Bayesova statistika u Pythonu

Radovniković, Vito

Master's thesis / Diplomski rad

2016

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Split, University of Split, Faculty of science / Sveučilište u Splitu, Prirodoslovno-matematički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:166:077667

Rights / Prava: Attribution-NonCommercial-ShareAlike 4.0 International/Imenovanje-Nekomercijalno-Dijeli pod istim uvjetima 4.0 međunarodna

Download date / Datum preuzimanja: **2024-04-18**

Repository / Repozitorij:

Repository of Faculty of Science

zir.nsk.hr

UNIVERSITY OF SPLIT

dabar

DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

SVEUČILIŠTE U SPLITU
PRIRODOSLOVNO MATEMATIČKI FAKULTET

DIPLOMSKI RAD

BAYESOVA STATISTIKA U PYTHONU

Vito Radovniković

Split, prosinac 2016.

Sadržaj

1. Uvod 4

2. Bayes-ov Teorem 5

2.1. Zajednička i uvjetna vjerojatnost 6

2.2. Analiza okruženja i prikupljanje podataka 10

2.3. Distribucija podataka 12

2.4. Definicija okvira distribucije 14

2.5. Začahurivanje okvira 17

2.6. Definiranje pretpostavki i izračunavanje procjena 18

2.7. Alternativni pristupi kod postavljanja pretpostavki 21

3. Napredne primjene Bayes-ovog teorema 23

3.1. Izračunavanje omjera 25

3.2. Analiziranje vjerojatnosti izbora 26

3.3. Interpretacija dokaza 30

4. Bayesova statistika u Pythonu : Euro problem 36

4.1. Optimizacija postupka 38

4.2. Beta distribucija 40

5. Zaključak 43

6. Literatura 44

7. Sažetak 46

8. Summary 47

1. Uvod

U drugom svjetskom ratu, odjel za ekonomsko ratovanje (engl. Economic Warfare Division) američke ambasade u Londonu je koristio statističku analizu za mnogo ratnih primjena. Jedna od tih primjena je bila procjena njemačke industrije i njihove proizvodnje tenkove i ostatka opreme. Saveznici su zarobili različite popise knjiga, inventara, opreme i arhiva koji su uključivali serijske brojeve karoserija i motora tenkova. Analiza ovih arhiva je pokazala da su proizvođači stavljali brojeve na tenkove u setovima od 100 brojeva, i da su sami brojevi unutar svakog od setova korišteni sekvencijalno, ali i da nisu iskorišteni svi brojevi. Upravo zbog toga je problem procjene njemačkih tenkova bio iznimno olakšan. Koristeći ove podatke, američki i britanski analisti su radili procjene koje su bile puno manje od prijašnjih procjena i procjena drugih obavještajnih službi. Poslije rata, sami njemački arhivi su pokazali da su procjene ovih analiza bile iznimno precizne. Koristile su se slične analize za gume, kamione, rakete i ostale dijelove ratne opreme, koje su olakšavale saveznicima ratno manevriranje.

Problem njemačkih tenkova je iznimno interesantan jer pokazuje iznimno bitnu primjenu preciznih statističkih analiza. Sama preciznost statistike, a pogotovo analize kroz Bayes-ove modele daje prednost i olakšava mnogim analizama i statističarima da olakšaju odluke i prijeđu u istraživačke vode. Kroz ovaj rad, predstavljati će se dijelovi osnovne problematike da bi se olakšao pristup Bayes-ovim modelima. Naravno, nakon osnovnih problema, logično je prijeći na srž bitnijih problema. Cilj ovog rada je predstaviti Bayes-ovu analizu i statističke metode njegovog teorema kroz različite problemske primjere koristeći biblioteke programskog jezika Python.

2. Bayes-ov Teorem

Thomas Bayes je bio statističar, filozof i prezbiterijanski svećenik. Rođen u Londonu sredinom 18.og stoljeća, Bayes se bavio raznim stvarima u svome životu, ali najpoznatiji je njegov utjecaj u polju statistike. Bayes-ova vjerojatnost je vjerojatnost koja se temelji na jačini vjerovanja u nekakve teorije i hipoteze, za razliku od dobrog dijela ostalih vjerojatnosti, koje se temelje na frekvenciji. Bayes nikada nije objavio svoj rad, već je Richard Price pronašao sve njegove zapise te ih lektorirao i objavio u Bayes-ovo ime. Sama Bayes-ova statistika je iznimno postala popularna 1950. godine i ostaje popularna sve do danas. Koristi se u mnoštvu različitih polja i od iznimne je važnosti pa takva i zauzima posebno mjesto u polju statistike. Poznavanje uvjetne vjerojatnosti olakšava snalaženje kroz svu Bayes-ovu statistiku, a samim time i Bayes-ov model na kojim se sve temelji. Zato se prvo započinje sa vjerojatnošću, pa zatim posebnim granama vjerojatnosti, Bayes-ovim modelom i na kraju završni pogled na Bayes-ovu statistiku [1].

Vjerojatnost je najčešće broj između 0 i 1 kojim se predstavlja stupanj mogućnosti za nekakvu činjenicu ili predikciju. Sam broj 1 predstavlja siguran događaj dok broj 0 predstavlja neistinu, laž ili nepostojanost događaja. Na primjer, bacanje novčića ima vjerojatnost od 0.5, što znači 50 %, odnosno da je vjerojatnost da će novčić pasti na glavu jednaka kolika i da padne na rep. Većina **uvjetne vjerojatnosti** (engl. Conditional probability) se temelji na nekakvim pozadinskim informacijama. Na primjer, ako želimo saznati vjerojatnost srčanog udara za prosječnog Amerikanca moramo provjeriti postojeće podatke. Ministarstvo zdravstva SAD-a tvrdi, po svojim zadnjim istraživanjima, da svake godine oko 785 tisuća Amerikanaca doživi svoj prvi srčani udar. Ako je populacija SAD-a oko 311 milijuna, mogućnost da slučajno odabrani Amerikanac doživi srčani udar je 0.3%. Ovu tvrdnju treba sagledati pomnije, jer postoji još faktora koji utječu na ovakav razvoj događaja.

Epidemiolozi su identificirali faktore koji utječu na srčani udar, a ti faktori su različiti kod svakog Amerikanca. Ovisno o kolesterolu, tlaku i korištenju cigareta, postotak se mijenja. Upravo zbog toga je potrebno poznavati što više pozadinskih informacija da bi se slika upotpunila. Uobičajena notacija za uvjetnu vjerojatnost je p(A | B) , što je vjerojatnost od A, ako je B istina. U ovom slučaju, A je postotak da će prosječni Amerikanac imati srčani udar, dok je B popis uvjeta kao tlak i kolesterol koji se razlikuju za svakog Amerikanca [11].

Zajednička vjerojatnost (engl. Conjoint probability), odnosno p(A AND B) zapravo označava vjerojatnost da su obje stvari istinite [12]. Formula za bacanje novčića i kockica iznosi:

$$P\ (\ A\ AND\ B\ ) = P(A)\ P\ (B) \qquad (1)$$

Na primjer, ako se bace dva novčića i postave oznake na način da ako prvi novčić padne na glavu se označi kao A, dok ako drugi novčić padne na glavu se označi sa B, onda vrijedi p(A) = p(B) = 0.5 čime dolazimo do toga da je p(A and B) = p(A) p(B) = 0.25. Ova formula funkcionira jer su A i B nezavisni jedan od drugog. Odnosno padanje prvog novčića ne mijenja vjerojatnost padanja drugog. Na koju god stranu padne prvi novčić, vjerojatnost padanja drugog novčića na bilo koju stranu je i dalje ista. Odnosno vrijedi:

$$P\ (\ B|\ A\ ) = P\ (B) \qquad (2)$$

Potrebno je pogledati primjer gdje događaju nisu nezavisni da bi se slika upotpunila. Pretpostavimo da će nam A predstavljati današnju mogućnost kiše, dok B predstavlja sutrašnju mogućnost kiše. Ako znamo da je danas padala kiša, veća je vjerojatnost da će padati i sutra pa stoga zaključujemo da p(B | A) > p(B), odnosno da je veća mogućnost kiše sutra (B) ako kiša pada i danas (A). Znači ako je postotak kiše za bilo koji dan 0.5, postotak kiše za dva dana jedan za drugim nije 0.25 nego malo veća. Generalno, zajednička vjerojatnost se može izraziti na sljedeći način, za bilo koje A i B:

$$P\ (\ A\ AND\ B\ ) = P(\ A\ )P(\ B\ |\ A\ ) \qquad (3)$$

2.1. Zajednička i uvjetna vjerojatnost

Postoje dvije zdjele kolačića, jedna ima 30 kolačića od vanilije i 10 čokoladnih kolačića, dok druga zdjela ima 20 od vanilije i 20 od čokolade. Zatim nasumice se izvlači jedan kolačić

koji je od vanilije. Potrebno je provjeriti validnost i vjerojatnost tvrdnje da je izvučeni kolačić iz zdjele broj 1.

Ovo je **uvjetna vjerojatnost**. Ako se želi izraziti p(Zdjela 1 | Vanilija), potrebno je prvo računati vjerojatnost kolačića od vanilije i da postoji vrijednost vjerojatnost Zdjele 1 lako bi izračunali se izračunala :

$$P\ (\ Vanilija \mid Zdjela1\ ) = 3/4 \tag{4}$$

Nažalost, ne vrijedi P (A | B) = P (B | A) pa je potrebno pronaći način kako dobiti jednu vrijednost iz druge. Ovo je izvrsna prilika da demonstriranje Bayes-ovog teorema [11].

Počinje se sa izrazom komutativnosti koji je izražen kroz objedinjenu vjerojatnost, za bilo koje događaje A i B:

$$P\ (\ A\ AND\ B\ ) = P(\ B\ AND\ A\ ) \tag{5}$$

Zatim se određuje vjerojatnost objedinjenosti :

$$P\ (\ A\ AND\ B\ ) = P(A)P(\ B \mid A\ ) \tag{6}$$

S obzirom na to da nije određeno što predstavljaju A i B, oni su izmjenjivi.

$$P\ (\ B\ AND\ A\ ) = P(\ B\ )\ P\ (\ A \mid B\ ) \tag{7}$$

To je sve što je potrebno za izraziti krajnju jednakost. Iz čega slijedi:

$$P\ (\ B\ )\ P\ (\ A \mid B)) = P(\ A\ )P(B \mid A\ ) \tag{8}$$

Ovo zapravo govori da postoje dva načina za izračunavanje objedinjenosti. Ako postoji vrijednost mogućnosti A (p (A)), potrebno ju je pomnožiti sa uvjetnom vjerojatnošću P(B | A) ili obratno.

Na kraju je potrebno izraz podijeliti sa P(B) [12].

$$P(A \mid B) = \frac{P(B \mid A)P\ (A\ )}{P(B)} \tag{9}$$

Time se dobije Bayes-ov teorem [1].

Bayes-ov teorem u statistici opisuje vjerojatnost nekakvog događaja, ovisno o uvjetima i drugim događajima koji su vezani za glavni događaj. Upravo ovaj teorem se može iskoristiti za riješiti problem kolačića [8]. Sa oznakom B_1 označavamo hipotezu da je kolačić došao iz zdjele 1, a sa V označavamo sami kolačić od vanilije. Koristeći Bayes-ov teorem dobijemo :

$$P\ (\ B1 \mid V\ ) = \frac{\mathrm{P}(\ \mathrm{B1}\ )\ \mathrm{P}\ (\ \mathrm{V} \mid \mathrm{B1}\ )}{\mathrm{P}\ (\ \mathrm{V}\ )} \tag{10}$$

S lijeve strane se nalazi vjerojatnost zdjele 1, u slučaju da je izvučeni kolačić baš kolačić od vanilije. S desne strane se razlikuje nekoliko stvari:

- P (B_1) – Vjerojatnost da će se iskoristiti zdjelu 1 , neovisna o tome koji se kolačić nasumično izvuče. Pošto je problem u tome što se izvlači kolačić iz nasumično odabranih zdjela, pretpostavka vrijednosti od P (B_1) iznosi 1 / 2.
- P (V | B_1) – Vjerojatnost da kolačić od vanilije bude izvučen iz prve zdjele, koja iznosi 3 / 4 .
- P (V) – Ovo je vjerojatnost da će se izvući kolačić od vanilije iz bilo koje zdjele. Pošto postoje jednake šansu odabrati bilo koju zdjelu, a zdjele sadržavaju isti broj kolačića, postoje iste šansu i odabrati bilo koji kolačić. Između dvije zdjele postoji

50 kolačića od vanilije i 30 kolačića od čokolade od čega proizlazi vjerojatnost P (V) = 5 / 8 .

Povezivanjem svih ovih izraza i njihovih vrijednosti, dolazi se do jednakost :

$$P\,(B1 \mid V\,) = \frac{(\,(\,1 / 2\,)\,(\,3 / 4\,))}{(\,5 / 8\,)} \tag{11}$$

Izraz s desne strane se reducira na 3 / 5 što zapravo govori da je kolačić vjerojatnije stigao iz zdjele broj 1 jer je i sama hipoteza postavljala pitanje o vjerojatnosti odabiranja zdjele broj 1.U ovakvim primjerima gdje se lakše izračunavaju izrazi vjerojatnosti s desne strane izraza nego sa lijeve strane izraza, ova strategija je iznimno korisna. Postoji još jedan način promatranja Bayes-ova teorema, a to je da pomoću njega se dolazi do vjerojatnosti hipoteze H u slučaju nekih podataka D. Ovakvo generaliziranje teorema se zove **dijakronijska interpretacija**. Dijakronijska znači da će se nešto događati kroz neko vrijeme, što znači da će se vjerojatnost hipoteze mijenjati ovisno o novim podacima kroz to vrijeme [3].

Ako ubacimo hipotezu H i podatke D u Bayes-ov teorem dobijemo :

$$P\,(H \mid D\,) = \frac{P(\,H\,)\,P(\,D \mid H\,)}{P\,(\,D\,)} \tag{12}$$

U ovoj interpretaciji svaka varijabla i vjerojatnost ima svoje ime.

- P (H) - Vjerojatnost hipoteze prije nego uopće se uopće koriste ikakve podatci. Ova vjerojatnost se stoga zove prior.
- P (H | D) – Vjerojatnost hipoteze nakon što sami podatci postanu poznati, odnosno cilj samog izračuna. Ovakve vjerojatnosti se nazivaju posteriori.
- P (D | H) – Vjerojatnost podataka pod hipotezom, nazvana ishod (engl. Likelihood).
- P (D) – Vjerojatnost podataka pod bilo kojom hipotezom, koja se često zove konstanta normale.

Ponekad se može izračunati prior zahvaljujući isključivo pozadinskim informacijama. Prikazani problem s kolačićima je specificirao da se mora izabrati zdjelu nasumično sa

jednakom vjerojatnošću. U nekim drugim slučajevima prior će biti subjektivan. Neki ljudi mogu drugačiji shvatiti pozadinske informacije ili ih mogu odlučiti iskoristiti na potpuno drugačiji način. Ishod je najčešće najlakša stvar za izračunati. Ako znamo odakle je stigao kolačić vanilije, može se lagano izračunati vjerojatnost jednostavnim prebrojavanjem. Konstanta normale zna biti malo kompliciranija. Ona bi trebala biti vjerojatnost da će postojati pristup podacima pod bilo kojom hipotezom, ali u većini slučajeva je teško odrediti što to točno znači [1].

Najčešće se pojednostavni cijelo određivanje vjerojatnosti tako što se odredi specifičan set hipoteza :

- **Zajedničko ekskluzivna** (engl. Mutually exclusive) – Najviše jedna hipoteza u cijelom setu može biti istinita.
- **Kolektivno izdašna** (engl. Collectively exhaustive) – Ne postoje druge mogućnosti, najmanje jedna hipoteza mora biti istinita.

U problemu sa kolačićima razlikuju se samo dvije hipoteze, da je kolačić došao iz zdjele 1 ili zdjele 2. Obe ove hipoteze su ujedno i zajedničko isključive i kolektivno izdašne. U tome slučaju je potrebno izračunati P (D) koristeći zakone vjerojatnosti koji iskazuju jednakost da postoje dva ekskluzivna načina da se dođe do događaja [7].

$$P(D) = \mathrm{P}(\mathrm{B1})\,\mathrm{P}(\mathrm{D} \mid \mathrm{B1}) + \mathrm{P}(\mathrm{B2})\,\mathrm{P}(\mathrm{D} \mid \mathrm{B2}) \tag{13}$$

Ako se iskoriste vrijednosti iz našeg problema o kolačićima, dolazi se do sljedećeg izraza :

$$P(D) = (1/2)(3/4) + (1/2)(1/2) = 5/8 \tag{14}$$

Što zapravo pokazuje istu stvar koja je dobivena ranijim izračunima.

2.2. Analiza okruženja i prikupljanje podataka

Postoje mnogi različiti poznati problemi koji se mogu riješiti koristeći Bayes-ov teorem. S toga potrebno je uzeti jedan poznati primjer da bi se prikazalo od kolike je važnosti zapravo

prikupiti i analizirati sve podatke. Jedan od njih je pitanje je vezano za Elvisa Presley-a. Elvis Presley je imao brata blizanca koji je umro pri porodu. Potrebno je izračunati vjerojatnost da je blizanac koji je preminuo bio jednojajčani blizanac [4]. Da bi se pristupilo rješavanju ovog problema, potrebno je sagledati sve pozadinske podatke. Pristupajući raznim medicinskim člancima, dolazi se do podatka da blizanci čine oko 1.9% svjetske populacije, dok identični blizanci čine 0.2% svjetske populacije odnosno 8% populacije svih blizanaca. Sada kada se prikupilo dovoljno pozadinskih podataka za početak, može se pristupiti rješavanju na par načina. Najlakši način je da se sagledaju rođenja blizanaca i odmah postavi hipoteza da je Elvis imao brata blizanca.

Razlikujemo sljedeće hipoteze :

- A – Rođenje je bilo dvoje jednojajčanih blizanaca
- B – Rođenje je bilo dvoje dvojajčanih blizanaca

Ako identični blizanci čine 8% svih blizanaca, to znači da su rođenja identičnih blizanaca 8% od svih rođenja blizanaca. Koristeći ove pozadinske informacije, vrijednosti priora glase:

- P (A) = 8%
- P (B) = 92%

Bitan podatak je bio i o spolu Elvisova blizanca.

- E – Elvisov blizanac je bio muškog spola.

Vjerojatnosti su sljedeće :

- P (E | A) = 1
- P (E | B) = 1 / 2

Ovo proizlazi iz toga što identični blizanci ne moraju nužno biti istoga spola, a dvojajčani blizanci imaju jednake šanse biti različitog spola.

Iz čega se dobije vjerojatnost :

$$P(A \mid E) == 8 / 54 \sim 0.15 \qquad (15)$$

Vjerojatnost da je Elvis imao identičnog blizanca je 0.15. Ključ korištenja Bayes-ovog modela i postavljanja cijele funkcije je u dobavljanju ključnih informacija, u ovom slučaju spol blizanaca i sami postotak identičnih blizanaca u svjetskoj populaciji.

Kada se razrješava problematika koja koristi uvjetnu vjerojatnost, Bayes uvelike olakšava posao sa svojom taktikom „podijeli pa vladaj“. Ako je preteško izračunati ili izmjeriti P (A | B) , onda se pristupa drugačije na način da se razmotri da li je lakše provjeriti ostatak u Bayes-ovom teoremu kao što su na primjer P (B | A), P (A) ili P (B) .

2.3. Distribucija podataka

U statistici postoje setovi vrijednosti i njihove vjerojatnosti koje nazivamo distribucijama. Na primjer ako se bacaju igraće kockice sa 12 strana, set mogućih vrijednosti su brojevi od 1 do 12, a vjerojatnost svake vrijednosti je 1 / 12.

Da bi se prikazalo korištenje distribucije u Python-u, koristiti će se klasu PMF (engl. Probability mass function) koja će olakšati prikazivanja raznih vrijednosti i njihovih vjerojatnosti. Ova klasa je razvijena i napravljena u svrhu knjige Think Bayes autora Allen B. Downeya. Pošto se čitava Bayes-ova analiza oslanja na statističke metode i izračunavanja vjerojatnosti, koristiti će se primjeri u sklopu ove biblioteke koji već imaju uključena razna statistička mjerenja i izračunavanja [10].

Da bi se moglo koristiti klasu PMF i sve njene metode, prvo je potrebno ju uključiti u sami projekt. Ova klasa koristi mnoštvo različitih metoda i implementacija koje će olakšati baratanje Bayes-ovom logikom. Kroz nju koristiti će se metode za osvježavanje podataka, postavljanje hipoteza, izračunavanje srednjih vrijednosti pa čak i distribucija podataka. Da se ne koristi ova biblioteka, već da se pokušavaju napisati ove sve metode iz početka, bilo bi potrebno jako mnogo matematičke pripreme i testiranja samih algoritama.

Za početak je potrebno unijeti sve vrijednosti da bi se prikazala distribucija vjerojatnosti za kocku sa 12 strana.

```
from thinkbayes import Pmf                    Kod 1
pmf = Pmf()
for x in
[1,2,3,4,5,6,7,8,9,10,11,12]:
      pmf.Set(x,1/12.0)
```

Implementacijom klase pmf se stvara prazan Pmf objekt bez ikakvih vrijednosti postavljenih vrijednosti na početku. Pomoću set metode, postavlja se vrijednost vjerojatnosti za svaku od vrijednosti unutar x-a (*Kod 1*). Korisno je sagledati način na koji se prebrojava koliko puta se koja riječ puta ponavlja u određenoj listi.

```
pmf = Pmf()                              Kod 2
for word in word_list:
      pmf.Incr(word,1);
```

Koristeći metodu Incr (engl. Increment) sama vrijednost ponavljanja za svaku određenu riječ se povećava za 1 . Ako navedena riječ ne postoji, ona se nadodaje i postavlja se na inicijalnu vrijednost 1 (*Kod 2*). U ovom slučaju ovi brojevi su proporcionalni sa vjerojatnošću ponavljanja ovih riječi. Nakon što se prebroje sve riječi potrebno je izračunati vjerojatnost tako što se podijeli vrijednost svake riječi sa ukupnim brojem riječi. Pmf klasa sadrži metodu „Normalize“ koja olakšava čak dijeljenje i normaliziranje brojeva. (*Kod 3*)

```
pmf.Normalize()                          Kod 3
```

Normaliziranje koristimo za lakše predstavljanje podataka u rasponu između 0 i 1. Pošto se čitava vjerojatnost temelji na brojevima između 0 i 1, u ovome slučaju je lakše držati sve podatke u istovjetnoj nomenklaturi [1].

2.4. Definicija okvira distribucije

Koristeći pmf klasu da bi se pridodalo svakoj vrijednosti njenu vjerojatnost olakšava iznimno kada se razlikuje mnogo hipoteza u Bayes-ovom teoremu. Distribucija koja sadrži priore za svaku hipotezu se zove **distribucija priora** [3], [2].

```
pmf = Pmf()                          Kod 4
pmf.Set('Zdjela1',
0.5)
pmf.Set('Zdjela1',
0.5)
```

Da bi se podaci osvježili i nadodali novi, kao što je na primjer kolačić od vanilije, potrebno je pomnožiti svaki od naših priora sa njegovom vjerojatnošću (*Kod 4*) Vjerojatnost da se izvučekolačić od vanilije iz Zdjele1 iznosi 3 / 4, dok za Zdjela2 iznosi 1 / 2) (*Kod 5).*

```
pmf.Mult('Zdjela1',                  Kod 5
0.75)
pmf.Mult('Zdjela2',
0.5)
```

Metoda Mult dohvaća vjerojatnost dane hipoteze i pomnoži ju sa faktorom ishoda , u ovom slučaju sa brojevima 0.5 za zdjelu 1 i 0.75 za zdjelu 2.

Nakon što se pomnoži dana vjerojatnost sa faktorom ishoda, potrebno je normalizirati samu distribuciju (*Kod 6).*

```
pmf.Normalize()                      Kod 6
```

Rezultat ove distribucije koja sadrži vjerojatnost posteriora za svaku hipotezu, a zove se distribucija posteriora (*Kod 7).*

```
print pmf.Prob('Zdjela1')
```
Kod 7

Dobijemo rezultat 0.6 .

Bilo bi dobro formulirati cijelu klasu da se bavi problemom kolačića. Koristeći ovaj primjer može se definirati i prikazati korištenje Bayes-ovog okvira (engl. Bayesian framework) . Prvo je potrebno definirati klasu koja će sadržavati sve potrebne stvari na početku.

```
class cookie(Pmf):
   def __init__(self, hypos):
      Pmf.__init__(self)
      for hypo in hypos:
            self.Set(hypo,1)
            self.Normalize()
```
Kod 8

Pomoću ove klase se stvaraju Pmf objekte klase Cookie. Prednost ovoga je što je olakšano pridodavanje vjerojatnosti svakoj od hipoteza. Metoda __init__ daje svakoj hipotezi jednaku vjerojatnost priora. *(Kod 8).* Razlikujemo dvije hipoteze kao i prije.

```
hypos = ['Zdjela1',
'Zdjela2']
pmf = Cookie(hypos)
```
Kod 9

Koristeći klasu Cookie mnogo je lakše doći i do metode za osvježavanje (engl. Update) koja traži podatak kao parametar, a zatim pomoću tog podatka osvježava vrijednost vjerojatnosti.

```
def Update (self, data ) :
 for hypo in self.Values():
  like = self.Likelihood(data,hypo)
            self.Mult(hypo,like)
 self.Normalize()
```
Kod 10

Metoda za osvježavanje prolazi kroz sve hipoteze i množi njihove vjerojatnosti sa vrijednosti ishoda svakog podatka unutar hipoteze (*Kod 10).* Do tih podataka se dolazi pomoću Likelihood metode (*Kod 11).*

```
mixes = { 'Zdjela 1':dict(vanilija =                    Kod 11
0.75, čokolada = 0.25),
            'Zdjela 2':dict(vanilija =
            0.5 , čokolada = 0.5),
                    }
def Likelihood(self, data, hypo):
            mix = self.mixes[hypo]
            like = mix[data]
            return like
```

Likelihood metoda koristi mixes, što je zapravo rječnik koji pridodaje svakoj zdjeli količinu kolačića. Potrebno je razmotriti pozivanje metode za osvježavanje: (*Kod 12).*

```
            pmf.Update('vanilija')                  Kod 12
```

Zatim se korisniku na ekranu može prikazati vjerojatnost posteriora za svaku hipotezu (*Kod 13).*

```
            for hypo, prob in                   Kod 13
            pmf.Items():
                  print hypo, prob
            ------------
            Zdjela 1 0.6
            Zdjela 2 0.4
```

Razmatranjem ovih podataka se dolazi do zaključka da se dobilo iste vrijednosti koje su bile izračunate imali i prije. Jedna od prednosti je ta što ovako se svodi hipoteze na slučaj kada se izvlači više od jednog kolačića iz iste zdjele (*Kod 14).*

Kod 14

```
dataset = ['vanilija',
'čokolada',
'vanilija']
for data in dataset:
      pmf.Update(data)
```

Druga prednost je u tome što definiranjem ove klase se dobije okvir za rješavanje različitih sličnih problema [1].

2.5. Začahurivanje okvira

Sada kada je jasno koji su dijelovi okvira identični, korisno bi bilo ih začahuriti (engl. Encapsulate) u objekt koji nazvan Suite. Ovakav objekt će sadržavati metode __init__, Update i Print.

Kod 15

```
class Suite(Pmf):

"Predstavlja cijeli set hipoteza i
njihove vjerojatnosti."

      def           __init__(self,
hypo=tuple()):

            „Pokretanje
distribucije."

      def Update(self, data):

            „Osvježava      podatke
svake od hipoteza:"

      def Print(self):

            „Prikazuje   svaku   od
hipoteza i njihove vjerojatnosti."
```

Sama klasa Suite se nalazi u datoteci thinkbayes.py , a da bi se mogla koristiti potrebno je napisati novu klasu koja će ju naslijediti i zadati varijablu ishoda (*Kod 15).*

Suite je apstraktni tip, što znači da definira sučelje koji bi se uvijek trebalo koristiti, ali otvorenog tipa gdje ga korisnik nadograđuje i nasljeđuje kako poželi. Suite sučelje se sastoji od metode za osvježavanje (engl.Update) i metode za izračunavanje ishoda (engl. Likelihood), dok sama klasa Suite koristi samo osvježavanje, ali ne ishod.

Konkretni tip (engl. Concrete type) je klasa koja proširuje apstraktnu klasu svojeg roditelja i nadoda metode koje nedostaju. Ovo je primjer predloška metodologije uzorka, odnosno dijela uzoraka koji se koriste u dizajniranju raznih uzoraka u programiranju (engl. Design patterns).

Većina primjera koji su prikazani će koristiti identičan uzorak. Za svaki problem će biti potrebno definirati novu klasu koja će proširiti Suite, naslijediti Update i definirati Likelihood metodu [1], [2], [3].

2.6. Definiranje pretpostavki i izračunavanje procjena

Ponekad različiti izračuni zahtijevaju postavljanje određenih pretpostavki da bi se uopće moglo doći do izračuna. Prije nego što se može ustanoviti vrijednost priora, potrebno je definirati kojim se sve podacima barata i što se sve može zaključiti kroz istraživanje podataka i logično razmišljanje. Sljedeći problem se bavi primjenom ovih dodatnih metoda u svrhu izračuna vjerojatnosti ishoda.

„ *Lokomotive su najčešće označene od 1 … N . Jedan dan dok šetamo uz prugu ugledamo lokomotivu sa brojem 60. Procijeni koliko lokomotiva ima ta tvrtka.* „

Ovaj problem lokomotive je preuzet iz knjige „Fifty Challenging Problem sin Probability With Solutions“ autora Fredericka Mostellera [4], [5], [6].

Početna opservacija je da ova tvrtka mora imati minimalno 60 lokomotiva. Da bi se iskoristio Bayes-ov teorem, ovaj problem je potrebno razdijeliti u manje module.

1. Potrebno je provjeriti sve podatke koji postoje o varijabli N, prije nego što se sagleda problematiku.

2. Za bilo koju vrijednost od N, potrebno je definirati sve vjerojatnosti da bi se lakše odredilo lokomotive koje su označene sa N.

Odgovor na prvo pitanje je u ovom slučaju prior, dok je odgovor na drugo pitanje ishod (eng. Likelihood) .

Ne postoji puno informacija na kojima se može bazirati prior, ali zato se može pretpostaviti da će N biti jednak bilo kojoj vrijednosti između 1 i 1000 (*Kod 16).*

```
hypos = xrange ( 1,                    Kod 16
10001)
```

Sve što nam sada treba je funkcija ishoda. Nakon što se pretpostavi da flota ima N lokomotiva, potrebno je definirati vjerojatnost da je lokomotivu koja je ugledana označena sa brojem 60. Ako se pretpostavi da postoji samo jedna tvrtka koja upravlja vlakovima i da postoje jednake šanse vidjeti bilo koju od njihovih lokomotiva, onda je šansa da se ugleda specifičnu lokomotivu jednaka 1 / N (*Kod 17).*

```
class Train ( Suite ) :                                    Kod 17
            def Likelihood(self, data, hypo):
                  if hypo < data:
                        return 0
                  else:
                        return 1.0/hypo
```

Sada se poziva metoda za osvježavanje podataka (*Kod 18).*

```
suite = Train(hypos)                                  Kod 18
suite.Update(60)
```

Ispisivanje svih hipoteza za svaku lokomotivu bi bilo predugačko za ispisati, ali može se zaključiti da su sve vrijednosti varijable N ispod 60 eliminirane. Ako je upravo viđena lokomotiva sa brojem 60, sigurno ne postoje manje od 60 lokomotiva.

Svejedno bi trebalo razmotriti mogućnost da je zbroj svih lokomotiva jednak 60. Šanse su iznimno male da je upravo viđena lokomotivu sa maksimalno velikim brojem, no je cilj biti što precizniji u pogađanju broju lokomotiva potrebno je i testirati broj 60.

Alternativni pristup je da se izračuna distribucija posteriora.

```
def Mean ( suite ) :                                    Kod 19
      total = 0
      for hypo, prob in suite.Items():
            total += hypo*prob
      return total
print Mean(suite)
```

Prosječna srednja vrijednost (engl.mean) posteriora je 333 što je zapravo dobar pokušaj ako je cilj minimizirati šanse za grešku (*Kod 19).*

Da bi se napredovalo u rješenju ovog problema moralo bi se postavljati pretpostavke, a neke od njih znaju biti poprilično banalne ili blesave. Na primjer, odlučili smo koristiti prior u rasponu između 1 i 1000 bez ikakvog opravdanja za maksimalan broj.

Kada bih netko išao pogoditi koliko tvrtka ima lokomotiva vrlo vjerojatno bi rekao da ima ili više ili manje od 1000 lokomotiva. S obzirom na to da poprilično se poprilično slobodnih ruku određuje ovu vrijednost, sama osjetljivost distribucija posteriora i o njoj ovisi. Izračun srednje vrijednosti posteriora je bio 333. Kada se postavi gornju granicu na 500 umjesto 1000, dobije se srednju vrijednost od 207. Ako postavimo gornju granicu na 2000, dobije se srednju vrijednost od 552.

Postoji samo jedan način da se doskoči ovom, a to je da se dohvati što više pozadinskih informacija koje bi olakšale postavljanje pretpostavki. Na primjer ako uz lokomotivu sa brojem 60, unesemo podatak da su viđene i dvije lokomotive sa brojevima 30 i 90 (*Kod 20).*

```
for data in [60,30,90]:                    Kod 20
      suite.Update(data)
```

Kada se iskoriste ovi novi podatke, dolazi se do novih srednjih vrijednosti.

- Gornja granica : 500 – Srednja vrijednost : 152
- Gornja granica : 1000 – Srednja vrijednost : 164
- Gornja granica : 2000 – Srednja vrijednost : 171

Razlike su vidljivo manje [4], [5], [6].

2.7. Alternativni pristupi kod postavljanja pretpostavki

Naravno, ponekad jednostavno ne postoji pristup novim informacijama ili ih možda jednostavno niti nema. U ovom slučaju, može se pokušati istražiti što više pozadinskih informacija o samome sustavu koji se promatra.

U većini industrija na svijetu i gospodarstava razlikujemo mnoštvo malih poduzeća, ponešto srednjih poduzeća i jednog do dva giganta u svojoj industriji. Ovaj zakon distribucije se zove **zakon moći** (engl. Power law) . Koristeći ovaj zakon može se zaključiti da ako postoji 1000 kompanija sa manje od 10 lokomotiva, također može postojati oko 100 kompanija sa 100 lokomotiva i 10 kompanija sa 1000 lokomotiva. Čak može postojati i jedna sa 10 000 lokomotiva [13]. Matematička formula koja se koristi kroz zakon moći govori da je broj kompanija sa određenom veličinom invertno proporcionalan veličini.

$$PMF(X) \infty \left(\frac{1}{x}\right)^{a} \tag{16}$$

PMF (X) ovdje označava vjerojatnost masovne funkcije varijable x, dok a predstavlja parametar jako blizu jedinici.

Koristeći zakon moći (*Kod 21)*, može se lakše doći do priora.

Kod 21

```
class Train(Dice):

    def __init__(self, hypos, alpha = 1.0 ):
            Pmf.__init__(self)
            for hypo in hypos:
               self.Set(hypo ,hypo**(-alpha))
                        self.Normalize()
```

Sami prior se izračunava na sljedeći način:

```
hypos = range(1, 1001)                                    Kod 22
      suite = Train(hypos)
```

Ovdje je opet uzet broj od 1000 za gornju granicu, dok zapravo se može uzeti bilo što proizvoljno i razumno u ovom slučaju jer rezultat neće biti pretjerano osjetljiv na gornju granicu zbog korištenja zakona moći (Kod 22).

Koristeći novi izračun priora i ubacivanjem podataka da su viđeni vlakovi pod brojem 30, 60 i 90, dobije se rezultat srednje vrijednosti koji je jednak sljedećem:

- Gornja granica : 500 – Srednja vrijednost : 131
- Gornja granica : 1000 – Srednja vrijednost : 133
- Gornja granica : 2000 – Srednja vrijednost : 134

Kao što se vidi iz priloženih rezultata, razlike su još i manje sada. Ako se ubaci što više većih gornjih granica, to srednja vrijednost sve više pokazuje na vrijednost 134. Nakon što se izračuna distribucija posteriora, potrebno je nekako sažeti rezultate u određenu broj ili približno u nekakav interval. Ako se koristi određen broj, dobro je koristiti srednju vrijednost kao što je već viđeno u prijašnjem primjeru, no ako se želi drugačiji označiti i koristiti intervale onda je dobro koristiti **intervale kredibiliteta** (engl. Credible interval). Kada se koristi intervale obično se razlikuju dvije vrijednosti između kojih se nalazi tražena vrijednost. Jednostavan način za izračunati ovaj interval je da se uzme vjerojatnost iz distribucije posteriora i vrijednosti koje pripadaju vjerojatnosti sa 5% i 95%.

Najčešće, ovakvi izračuni mogu pokazati koliko je dobro ili loše naše rješenje u postotku, odnosno koliko je nesiguran rezultat razmatrane problematike [1], [2], [3], [13].

3. Napredne primjene Bayes-ovog teorema

Do sada se koristio raspon između 0 i 1 da bi se predstavila vjerojatnost nekakvog događaja. Naime, ovaj pristup se ne koristi uvijek u svim primjenama. U raznim kladionicama se koriste omjeri i koeficijenti da bi se odredila vjerojatnost nekakvog događaja. U zapadnjačkom svijetu se uvelike koriste omjeri kod klađenja, gdje se mogu čuti stvari kao „omjer je 3 naprama 1“. Kada su omjeri za događaj (engl. Odds in favor) to zapravo prikazuje odnos između vjerojatnost da će se nešto dogoditi i vjerojatnosti da se nešto neće dogoditi. Ako se pretpostavi da Hajduk ima 75% šanse da pobijedi u jednoj utakmici, to se može izraziti na način da su zapravo omjeri u korist Hajduka u odnosu 3 naprama 1 , odnosno 3:1. Kada se želi prikazati iznimno male vjerojatnost, odnosno događaje koje je vjerojatnost jako mala da će se dogoditi , onda se koriste omjere protiv događaja (engl. Odds against). Na primjer, ako Dinamo ima 10% šanse da pobijedi, omjeri protiv su 9:1 [1], [8]. Vjerojatnost i omjeri su različiti predstavnici istih informacija. Ako se dobije nekakvu vjerojatnost, nju je moguće izraziti kroz omjer na sljedeći način:

```
def Odds(p):                          Kod 23
      return p/(1-p)
```

Ako se dobije nekakve omjere za događaj u decimalnom obliku, oni se mogu izraziti kroz izračunavanje vjerojatnost na sljedeći način :

```
def Probability(o):                   Kod 23
      return o / (o+1)
```

Ako je na stadionu određen broj ljudi, i 20 % njih misli da će Hajduk pobijediti, to znači da 80% ljudi misli da Hajduk neće pobijediti. Pod uvjetom da je ishod utakmice ili pobjeda ili poraz, dobiju se omjeri za događaj jednake 20:80 odnosno 1:4. Ako su omjeri 5:1 za Dinamo, to znači da 5 od 6 ljudi u publici na Maksimiru misli da će Dinamo izgubiti, iz čega proizlazi vjerojatnost pobjede 1 / 6.

U drugom poglavlju Bayes-ov teorem je bio definiran obliku :

$$P\,(\,H \mid D\,) = \frac{P(\,H\,)\,P(\,D \mid H\,)}{P\,(\,D\,)} \tag{17}$$

Ako se razlikuju dvije hipoteze, A i B, može se izraziti i omjer vjerojatnosti posteriora.

$$\frac{P(\,A \mid D\,)}{P(\,B \mid D)} = \frac{P(\,A\,)\,P(\,D \mid A\,)}{P(\,B\,)\,P(\,D \mid B\,)} \tag{18}$$

U ovom slučaju, konstanta normalizacije P(D), više ne igra ulogu u izračunu. Ako postoje hipoteze A i B koje su međusobno isključive i kolektivno izdašne, to znači da je P(B) = 1 – P(A).

Ako je cilj zapisati mogućnost ishoda za omjere za događaj A, onda se dobije ovakav izraz:

$$o(\,A \mid D\,) = o(\,A\,)\frac{P(\,D \mid A\,)}{P(\,D \mid B\,)} \tag{19}$$

Iz ovoga zapravo proizlazi forma omjera Bayes-ovog teorema. Posterior omjera je jednak umnošku priora omjera i odnosu ishoda. Ova forma je korisna ako se želi brzo izračunati manje problematike koji su izraženi kroz omjere. Vratimo se na problem kolačića od vanilije. Razlikujemo dvije zdjela kolačića od vanilije. Zdjela 1 sadrži 30 kolačića od vanilije i 10 kolačića od čokolade. Zdjela 2 sadrži 20 kolačića od vanilija i 20 kolačića od čokolade. Ako se izvlači jedan kolačić od vanilije, koja je vjerojatnost da je došao iz zdjele broj 1. Vjerojatnost priora je 50%, što znači da je omjer priora 1:1. Odnos ishoda je ¾ / ½ , odnosno 3 / 2 . Omjer posteriora iznosi 3 : 2 što je zapravo vjerojatnost od 3 / 5 [1], [6], [8].

3.1. Izračunavanje omjera

Navedeni omjeri se ne koriste uvijek u svrhu igara na sreću. Ponekad se ovakve statističke metode koriste i u ozbiljnijim primjenama kao što je forenzika i kriminalistika. Kroz primjer mjesta ubojstva potrebno je sakupiti opet sve potrebne pozadinske podatke i informacije, a zatim pomoću omjera razriješiti dilemu.

„*Uzorci krvi dviju osoba su pronađeni na mjestu ubojstva. Oliver, jedan od osumnjičenika, je bio testiran i ima krv grupe 'O', dok je jedan od uzoraka sa mjesta ubojstva također potvrđena kao primjerak iz grupe 'O', a drugi iz krvne grupe 'AB'. Grupa 'O' je čest tip krvne grupe koju ima čak 60% populacije ju ima, a krvna grupa 'AB' je poprilično rijetka i ima je samo 1% populacije. Da li ovi podaci dokazuju da je Oliver bio na mjestu ubojstva.*“

Da bi se odgovorilo na ovo pitanje, prvo je potrebno sagledati sve podatke i dokaze za navedenu hipotezu. U prijašnjem problemu sa kolačićima, omjeri za prior su iznosili 1:1 odnosno postotak od 50%. Posterior omjeri su bili 3:2 odnosno 60%, iz čega proizlazi zaključak da su dokazi naginjali prema zdjeli broj 1 [8].

$$o(A \mid D) = o(A)\frac{P(D \mid A)}{P(D \mid B)} \tag{20}$$

Ako ovaj izraz od prije se podijeli sa o(A), dolazi se do sljedećeg izraza.

$$\frac{o(A \mid D)}{o(A)} = \frac{P(D \mid A)}{P(D \mid B)} \tag{21}$$

Lijevi dio jednadžbe je zapravo omjer odnosa posteriora i priora, dok je desni dio jednadžbe omjer ishoda koji nazivamo Bayes-ov faktor. Ako je Bayes-ov faktor veći od 1, to znači da su podaci podupirali više A nego B i samim time omjeri koji favoriziraju hipotezu B padaju. Dok ako je Bayes-ov faktor manji od 1, to znači da su podaci manje podupirali A nego B, što znači da omjeri koji favoriziraju hipotezu A padaju.

Preostao je samo slučaj kada je faktor jednak 1. To znači da podaci jednako mogući odnosno da jednako podupiru obje hipoteze, pa se zbog toga omjeri ne mijenjaju. Vratimo se na problem krvi. Ako je Oliver jedan od ljudi koji su ostavili krv na mjestu ubojstva, onda je

vjerojatnost podataka je zapravo vjerojatnost da je nasumičan prolaznik imao krvnu grupu 'AB' što je 1%. Ako Oliver nije ostavio krv na mjestu ubojstva, onda razlikujemo dva uzorka krvi koja su moramo nekome dodijeliti. Izvuku li nasumično dvije osobe iz cijele populacije, potrebno je definirati kolika je vjerojatnost da će se pronaći jednu sa jednu sa krvnom grupom 'O' i jednu sa krvnom grupom 'AB'. Prva osoba može imati krvnu grupu 'O' ili krvnu grupu 'AB', ali isto vrijedi i za drugu osobu. Konačna vjerojatnost za ovakav ishod je 2(0.6)(0.01)=1.2% [7]. Ishod ovih podataka je malo veći ako Oliver nije jedna od osoba koja je ostavila krv, što zapravo govori da su svi prikupljeni podaci o krvi dokaz krivnje Olivera. Ovaj primjer je malo zbunjujući zbog toga što je primjer neintuitivnog rezultata podataka koji su konzistentni sa hipotezom , koji ne moraju favorizirati hipotezu.

Ovo je moguće sagledati i iz drugog kutka. Podaci sadrže običan događaj, što je u ovom slučaju krvna grupa 'O' , i rijedak događaj odnosno krvnu grupu 'AB'. Ako je Oliver kriv za običan događaj, onda rijedak događaj ostaje neobjašnjen. Dok ako Oliver nije kriv za uzorak krvne grupe 'O' na mjestu ubojstva, onda imamo dvije šanse da će se izvući nasumično nekoga u populaciji sa krvnom grupom 'AB', i upravo je taj faktor od dvije šanse odlučujući [1], [4], [5], [7], [8].

3.2. Analiziranje vjerojatnosti izbora

Jedno od najčešće promatranih problema u polju vjerojatnosti je zasigurno Monty Hall. Odgovor na ovaj problem je toliko jednostavan, ali nije niti malo intuitivan što zna stvarati popriličnu zbunjenost. Monty Hall je bio voditelj emisije „*Let's Make a Deal"*. Problem koji se promatra je baziran na jednoj od manjih igrica u sklopu cijele emisije. Sama emisija se sastojala od publike, natjecatelja i voditelja emisije. Iz publike, koja je sudjelovala u cijelom formatu natjecanja, bi se odabrao određen broj ljudi koji bi postali natjecatelji koji su se zvali trgovci (engl. Traders) i njihov je cilj bio da sklapaju dogovore sa voditeljem emisije. U većini slučajeva, trgovcu bi se ponudilo nešto što bi imali nekakvu vrijednost, a trgovac bi birao da li će zadržati taj objekt ili ga zamijeniti za neki drugi objekt. Glavna mehanika ove igre se temeljila na tome da je taj drugi objekt bio sakriven sve dok trgovac ne odluči da li želi novi objekt ili želi zadržati stari. Zbog toga trgovac na slijepo zapravo bira sljedeći objekt koji može biti puno veće vrijednosti ili puno manje vrijednosti [5], [8].

Sami problem koji se pokušava razriješiti je baziran na samome biranju objekata i nazvan je po voditelju emisije Monty Hall-u. Ovaj problem je originalno postavljen 1975. godine u dnevniku *American Statistician* od kada je postao poznati problem i omiljen među statističarima.

Ovo su osnovne postavke problema.

1. Monty Hall u svojoj emisiji prezentira različita zatvorena vrata i kaže da se nalazi nagrada iza svakih od vrata. Jedna od nagrada je iznimno vrijedna, a to je automobil. Ostale dvije nagrade su bile najčešće trivijalne, kao što je na primjer fen za kosu, čaša vode, lak za nokte i slično. Sve tri nagrade su nasumično postavljene iza vrata.
2. Cilj igre je pogoditi gdje se nalazi automobil, jer ako natjecatelj uspije pronaći automobil odmah ga i osvaja.
3. Natjecatelj bira koja vrata želi prvo otvoriti. Na primjer, vrata A odnosno prva vrata.
4. Prije nego se otvore vrata koja je natjecatelj odabrao, Monty pokušava podignuti napetost jer je to ipak emisija, tako što otvara prvo vrata B ili C (otvoriti će ona koja ne sadrže automobil). Ako je automobil iza vrata A onda nasumično otvara vrata B ili C.
5. Zatim Monty nudi natjecatelju mogućnost da odabere druga vrata koja su neotvorena ili ostane uz svoj prvotni odabir koji je u ovom slučaju bio vrata pod oznakom A.

Glavni dio problematike se odnosi na to da li se isplati ostati na svom prvotnom izbori ili bi natjecatelj trebao promijeniti svoj izbor. Potrebno je sagledati da li uopće takav izbor ima ikakvog utjecaja na ishod cijele igre.

Većina ljudi tvrdi da nema nikakve razlike ni utjecaja na ishod. Ako postoje još dva vrata od kojih možemo odabrati jedna, pa su šanse da je automobil ispred vrata A koja smo prvotno odabrali jednaka 50%. No to je potpuno krivo. Stvarne šanse da se pobijedi ako i dalje ostanemo na vratima sa oznakom A su 1/3, dok ako promijenimo svoj odabir šanse za pobjedu su nam 2/3. Koristeći Bayes-ov teorem, možemo modulirati ovaj problem razbijanjem u više manjih komadića i možda uspijemo uvjeriti sami sebe da je točan odgovor zapravo točan, koliko god bio neintuitivan.

Za početak, potrebno je sve pozadinske informacije i podatke koje se nalaze na raspolaganju u rješavanju ovog problema. U ovom slučaju podatak D se sastoji od dva dijela:

- Monty bira vrata B.
- Auto se ne nalazi iza vrata B.

Zatim se postavljaju postaviti hipoteze koje se promatraju.

- Hipoteza A – Automobil se nalazi iza vrata sa oznakom A.
- Hipoteza B – Automobil se nalazi iza vrata sa oznakom B.
- Hipoteza C – Automobil se nalazi iza vrata sa oznakom C.

Zatim je potrebno sagledati koji sve podatci se mogu sakupiti i što se može zaključiti na početku (Tablica 1)

	Prior P(H)	Ishod P(D | H)	P(H) P(D | H)	Posterior P(H | D)
A	1/3	1 /2	1/6	1/3
B	1/3	0	0	0
C	1/3	1	1/3	2/3

Tablica 1. Prikaz hipoteza i vrijednosti parametara

Pošto su nagrade postavljene nasumično iza svakih od vrata, ili barem tako bi trebale biti, određivanje vjerojatnosti priora nije teško. Automobil ima jednaku vjerojatnost da će biti iza svakih od vrata.

Da bi se odredio ishod podataka ovisno o svakoj od hipoteza je malo teži za postaviti.

- Ako je automobil iza vrata sa oznakom A, Monty može otvoriti bilo koja vrata B ili C bez straha da će otvoriti vrata sa automobilom. Naravno, Monty u svakom trenutku zna gdje se točno nalazi automobil. Pošto je automobil iza vrata A, vjerojatnost da automobil nije iza vrata sa oznakom B iznosi 1.
- Ako je automobil iza vrata sa oznakom B, Monty mora otvoriti vrata sa oznakom C, što znači da je vjerojatnost da će on otvoriti vrata B jednaka 0.

- Ako je automobil iza vrata sa oznakom C, Monty mora otvoriti vrata sa oznakom B, što znači da je vjerojatnost da će otvoriti vrata B jednaka 1, i šansa da tu pronađe automobil je jednaka 0.

Kada se zbroji treći stupac u tablici dobijemo vrijednost ½, odnosno kada se zbroje umnošci svih hipoteza sa njihovim ishodom. Zatim je potrebno razmotriti sve dobivene vrijednosti posteriora :

$$\mathrm{P(A \mid D)} = 1/3 \qquad (22)$$

$$P(C \mid D) = 2/3$$

Sam posterior dokazuje činjenicu da bi se trebalo promijeniti svoj odabir i odabrati vrata C. Postoje mnoge varijacije u ovome problemu. Jedna od prednosti Bayes-ovog teorema je što generalizira sve varijacije i postavlja ih u jedan veliki problem. Na primjer, ako se pretpostavi da će Monty uvijek odabrati vrata sa oznakom B ako mu to dopuštaju prilike, on će odabrati vrata sa oznakom C jedino ako to baš mora odnosno ako se automobil nalazi iza vrata sa oznakom B. Potrebno je razmotriti kako se mijenjaju podaci u tablici vjerojatnosti ako se uračuna i ova pretpostavka (Tablica 2.).

	Prior P(H)	Ishod P(D | H)	P(H) P(D | H)	Posterior P(H | D)
A	1/3	1	1/3	1/2
B	1/3	0	0	0
C	1/3	1	1/3	1/2

Tablica 2. Prikaz hipoteza i vrijednosti parametara

Jedina stvar koja se promijenila je vjerojatnost za hipotezu A, odnosno P(D | A). Ako se automobil nalazi iza vrata sa oznakom A, Monty može odabrati bilo koja vrata sa oznakama B ili C. U ovom slučaju se uzelo pretpostavku da Monty uvijek bira vrata sa oznakom B, iz čega proizlazi izraz:

$$P\ (\ D \mid A\ ) = 1 \qquad (23)$$

Rezultat ovoga sve ga je da su ishodi jednaki za hipotezu da se automobil nalazi iza vrata pod oznakom A i pod oznakom C, iz čega proizlazi da je vrijednost posteriora jednaka za obje hipoteze.

$$P\,(\,A \mid D) = P\,(\,C \mid D\,) = 1\,/\,2 \tag{24}$$

U ovom slučaju, činjenica da Monty uvijek bira vrata pod oznakom B nam ništa ne govori o tome gdje se zapravo nalazi automobil, što znači da u ovom slučaju uopće nema nikakve razlike da li će natjecatelj ostati uz svoj originalni odabir ili će ga promijeniti. Dok u slučaju da je Monty odabrao vrata sa oznakom C, znali bi da je vjerojatnost da se automobil nalazi iza vrata sa oznakom B jednaka 1.

Postoje varijacije ovog problema u kojima nema razlike u odabir samog natjecatelja, no postoji mnogo više varijacija u kojima je bolje promijeniti svoj odabir. Iz čega proizlazi da je najbolje uvijek promijeniti svoj odabir, jer niti jedna varijacija ne govori da je jedino dobro ostati uz svoj prvotni odabir [1], [7], [8].

3.3. Interpretacija dokaza

U Prirodoslovno-matematičkoj gimnaziji u Split, poznata još i pod nazivom MIOC, održano je posebno natjecanje između dva kandidata. Poznati fakultet MIT iz američkog grada Massachusetts održava natjecanje između dva najbolja učenika ove splitske gimnazije da bi popunili mjesto u svojem akademskom kadru studenata. Ova dva najbolja učenika se zovu Ana i Mirko i poprilično su slični u svojim kvalifikacijama, ali sa jednom ključnom razlikom. Na samome testu iz matematike Ana je dobila bolju ocjenu od Mirka.

Sam test, na uzoru na američke SAT testove, sadrži različite zadatke koji moraju procijeniti koliko je jedna osoba sposobna prebroditi razne zadatke na akademskoj razini. Test ima maksimalno 800 bodova, a naš je zadatak da pogledamo sve dokaze i odredimo da li je razlika u osvojenim bodovima dovoljan dokaz da je Ana bolje pripremljena od Mirka. Ana je osvojila 780 bodova, a Mirko je osvojio 740 bodova. Zapravo, oba rezultata su iznimno dobra što znači da bi Mirko i Ana bili odlični kandidati za popunu mjesta na ovom elitnom sveučilištu. U ovakvom slučaju bi osoba koja bira kandidate vrlo vjerojatno odabrala kandidata koji posjeduje bolje kvalitete u drugim poljima, odnosno niti jedan kandidat ne bi bio eliminiran na temelju ovog ispita. Pošto je potrebno primijeniti Bayes-ov teorem i čitavu

analizu, testirati će se samo hipotezu u kojoj se propitkuje koliko je Ana bolje pripremljena od Mirka na temelju ovog ispita.

Da bi se uopće mogli odgovoriti na ovo pitanje, još jednom je potrebno obaviti istraživanje i pripremiti sve pozadinske informacije o samome ispitu. U pravilu, sva pitanja na ovakvim ispitima su različitih težina jer samo testiranje daje bolje rezultate i procjenu sposobnosti kandidata. Provedeno je nekoliko testiranja i zaključeno je da iako različita pitanja su različitih težina, sam ispit je balansiran na takav način da i dalje nije bitno na koja pitanja je odgovorio koji kandidat već je generalni rezultat najbolji pokazatelj kandidatove sposobnosti. Pošto je nebitno da li su pitanja jednako teško zbog kvalitete samog ispita, postavlja se karakteristiku jedinstvenu za svakog kandidata koja će predstavljati vjerojatnost da je svako pojedinačno pitanje odgovoreno istinito. Ovu karakteristiku će se nazvati p_correct. Iako ovakvo pojednostavljivanje na prvu ostavlja dojam da rezultati neće biti pravilni, ovdje je potrebno vjerovati da su autori ispita u pravu kada kažu da je ispit savršeno balansiran na način da je samo krajnji rezultat bitan u prikazu nečijih sposobnosti.

Baš zbog toga što su ispiti balansirani, lakše će se izračunati i same ishode rezultata. Da bi se postavilo same modele i hipoteze, potrebno je još malo istražiti pozadinu ispita i cijelog sustava ispitivanja. Svaki kandidat dobije finalnu ocjenu koja prikazuje njegov broj točnih i netočnih odgovora. Ta ocjena se onda prebacuje u mjernu skalu u rasponu od 200 do 800. Kroz godine, format od 54 pitanja se koristio za održavanje ovog matematičkog ispita. Sama ocjena koji bi svaki kandidat dobio se temeljila na točnim odgovorima od kojih bi se oduzelo 0.25 bodova za svaki netočni odgovor.

Odbor koji odobrava ove testove, svake godine objavljuje distribuciju i ocjene kandidata kroz skalu od 200-800 bodova. Ako se ove ocjene pretvori u prvotni oblik prije nego što se oduzmu netočni bodovi, dolazi se zapravo do vrijednosti varijable p_correct što je broj točno odgovorenih pitanja. Može se koristiti distribuciju ovih originalnih ocjena da bi se modelirala distribucija točnih odgovora, odnosno varijable p_correct. Za to je potrebno napisati kod (*Kod 24*) koji će učitavati i procesuirati podatke.

```
class Exam(object):                                                    Kod 24
    def __init__(self):
      self.scale = ReadScale()
      score = ReadRanks()
      score_pmf = thinkbayes.MakePmfFromDict(dict(scores))
      self.raw = self.ReverseScale(score_pmf)
      self.prior = DivideValues(raw, 54)
```

U samoj klasi exam kroz inicijalizaciju začahurujemo informacije koji su prikupljeni o samome ispitu. Metode ReadScale() i ReadRanks() prolaze kroz podatke i vraćaju objekt koji će sadržavati same podatke. Samu varijablu self.scale se koristi za konvertiranje podataka iz originalnih vrijednosti u vrijednosti za ljestvicu od 200 do 800 i obratno. Sama lista scores je iskorištena kao lista za parove vrijednosti rezultata i njihovih frekvencija. Unutar score_pmf spremamo pmf objekt sa skaliranim ocjenama (skala od 200 do 800), dok self.raw će sadržavati pmf objekt sa originalnim ocjenama. Unutar self.prior se pospremaju svi pmf objekti sa vrijednošću varijable p_correct. Zatim se postavlja klasu Suite za svakog kandidata koja će predstavljati distribuciju točnih odgovora, odnosno varijable p_correct (*Kod 25*) [1], [8].

```
class Sat(thinkbayes.Suite):                                           Kod 25
      def __init__(self, exam, score):
            thinkbayes.Suite.__init__(self)
            self.exam = exam
            self.score = score
            #Distribucija priora
            for p_correct, prob in exam.prior.Items():
                  self.Set(p_correct, prob)

            #Osvježi podatke o rezultatima ispita
            self.Update(score)
```

Kada se stvaraju sami objekti odnosno kada se inicijaliziraju i postavljaju osnovne vrijednosti, uzimaju se sami objekti ispita (engl. Exam) i rezultat. Stvara se kopija distribucije kroz jednu petlju i zatim se osvježavaju podatci o samom ispitu ovisno o rezultatima ispita. Kao i do sada, nasljeđuje se metodu za osvježavanje iz klase Suite i posebno se definira metodu za ishod (*Kod 26)*.

Kod 26

```
def Likelihood(self, data, hypo):
      p_correct = hypo
      score = data

      k = self.exam.Reverse(score)
      n = self.exam.max_score

      like = thinkbayes.EvalBinomialPmF(k, n, p_correct)
      return like
```

Postavljanjem hipotetske vrijednosti unutar varijable p_correct, dok skalirane podatke sa ljestvice se pospremaju u varijablu data. Da bi se pojednostavila stvar još malo, originalni rezultat sa ispita se sprema kao broj točnih odgovora tako da se ignorira kaznu koja se dobije za netočne odgovore. Ishod koji ovako dobijemo korištenjem binomne distribucije izračunava vjerojatnost od k točnih odgovore od n pitanja.

Potrebno je razmotriti koliko su jaki dokazi da je Ana bolje pripremljena od Mirka. Za ovo je potrebno iskoristiti distribuciju posteriora za varijablu p_correct da bi se precizno odgovorilo na ovo pitanje. Zatim, postavljaju se dvije hipoteze:

- Hipoteza A : p_correct je veći kod Ane, nego kod Mirka.
- Hipoteza B : p_correct je veći kod Mirka, nego kod Ane.

Da bi se izračunao ishod hipoteze A, potrebno je numerirati sve parove vrijednosti iz distribucije posteriora i nadodati vjerojatnost slučajeva gdje je p_correct veći kod Ane nego kod Mirka. Već postoji funkcija unutar biblioteke ThinkBayes koja pomaže u ovome. Definiranjem objekta Suite koji će izračunati vjerojatnost posteriora za hipoteze A i B (*Kod 27*).

```
class TopLevel(thinkbayes.Suite):                                    Kod 27
      def Update(self,data):
            a_sat, b_sat = dana

            a_like = thinkbayes.PmfProbGreater(a_sat, b_sat)
            b_like = thinkbayes.PmfProbLess(a_sat, b_sat)
            c_like = thinkbayes.PmfProbEqual(a_sat, b_sat)

            a_like += c_like / 2
            b_like += c_like / 2

            self.Mult('A', a_like)
            self.Mult('B', b_like)

            self.Normalize()
```

Kada se inicijalizira novi Suite obično se nasljeđuje metoda za osvježavanje i napiše novu metodu za ishod. U ovom se slučaju premostila (engl. Override) metodu za osvježavanje jer je lakše procijeniti ishode obje hipoteze u isto vrijeme. Sami podaci koji se šalju kroz metodu za osvježavanje su sat objekti koji sadrži distribucija posteriora točnih odgovora, odnosno varijable p_correct. Varijabla a_like će sadržavati vjerojatnost da je p_correct za Anu veći nego za Mirka, dok će b_like sadržavati vjerojatnost da je p_correct veći za Mirka nego za Anu.

Varijabla c_like predstavlja mogućnost i vjerojatnost da su jednaki, i potrebno ju je koristiti jer se odabralo modelirati p_correct varijablu sa setom diskretnih varijabli. Ako se koristi više vrijednosti, c_like će biti manji, dok ako je p_correct neprekidan c_like će biti jednak nuli. Varijablu c_like se tretira kao varijablu koja mjeri stupanj greške i potrebno ju je razdvojiti jednako između a_like i b_like varijabli.

```
exam = Exam()
a_sat = Sat(exam, 780)
b_sat = Sat(exam, 740)


top = TopLevel('AB')
top.Update((a_sat, b_sat))
top.Print()
```

Kod 28

Ishod hipoteze A će iznositi 0.79, dok ishod hipoteze B iznosi 0.21. Sami omjer ishoda , odnosno Bayes-ov faktor koji se već spominjao nekoliko puta, iznosi 3.8 što znači da su ovi rezultati dokaz da je Ana bolja od Mirka u rješavanju ovog specifičnog testa. Ako je postojala ideja da su hipoteze A i B jednako moguće, sada bi trebali barem biti nevažeća kad se razmotre podaci da je vjerojatnost hipoteze A jednaka 79% , što znači da postoji 21% šanse da je Mirko bolje pripremljen od nje [4], [5] .

4. Bayesova statistika u Pythonu : Euro problem

Postoji mnoštvo poznatih problema koji su iznimno popularni u statističarskim krugovima, ali i amaterima koji uživaju u matematičkim problemima. Jedan od ovakvih poznatih problema je i Euro problem, koji je postavljen prije nekoliko desetaka godina. Kroz rješavanje ovog problema prezentirati će se problematika koja je objašnjena do sada, ali na jedan opširniji način.

„Ako zavrtimo jedan euro na stolu točno 250 puta, on pada na glavu 140 puta , a na rep 110 puta. Barry Blight, statističar iz Londona, je ustanovio da je ovo vrlo čudno, jer ako je kovanica nepristrana onda ovakav ekstremni rezultat ima vjerojatnost manju od 7 %.“

Kada se sagleda ovaj događaj i cijelu situaciju, nameće se pitanje da li je kovanica pristrana ili pošteno balansirana [6].

Da bi se odgovorilo na ovo pitanje prvo je potrebno odrediti vjerojatnost da će novčić pasti na glavu. Druga stvar koji se mora izračunati je vrijednost da li podaci potvrđuju hipotezu da je novčić nepristran. Svaki novčić ima vjerojatnost x da će pasti na glavu kada ga se zavrti. Čini se razumno da vrijednost varijable x ovisi o fizičkim karakteristikama novčića, pogotovo o njegovoj težini. Ako je novčić savršeno balansiran x bi trebao biti blizu 50%, ali ako novčić ima neravnomjernu težinu vrijednost varijable x će biti drugačija. Prvo se mora definirati osnovnu hipotezu. Hipoteza H_x će predstavljati vjerojatnost da novčić padne na glavu, odnosno x% za vrijednost između 0 i 100. Za ovakav zadatak koristiti će se uniformirani prior gdje je vrijednost od H_x jednaka za sve vrijednosti varijable x.

Funkcija ishoda je laka za napraviti u ovom slučaju. Ako je H_x istinita, vjerojatnost da će novčić pasti na glavu je jednaka X / 100 , dok je vjerojatnost da će pasti na rep jednaka 1-X/100 (Kod 29).

Kod 29

```
class Euro(Suite):
      def Likelihood(self, data, hypo):
            x = hypo
            if dana == 'G':
                  return x/100.0
            else:
                  return 1-X/100.0
```

Zatim se stvara objekt Suite i osvježiti podatke (Kod 30).

```
suite = Euro(xrange(0,101))                                    Kod 30
dataset = 'G' * 140 + 'R' * 110
for data in dataset :
      suite.Update(data)
```

Postoji razni načini da se sumira distribucija posteriora. Jedan način je da se pronađe najvjerojatniju vrijednost u distribuciji, a za to čak postoji i funkcija (*Kod 31).*

```
def MaximumLikelihood(pmf):                                    Kod 31
      „Vraća vrijednost za najvećom vjerojatnošću.“
      prob, val = max((prob, val) for val, prob in pmf.items())
      return val
```

Ovako se dolazi do rezultata 56, što je zapravo i postotak padanja na glavu (140 / 250 = 0.56%). Ovo zapravo govori da je dobivena vrijednost zapravo i maksimalan ishod za cijelu populaciju. Ovu funkciju je lakše koristiti tako da se izračunaju i srednje vrijednosti (*Kod 32).*

```
print 'Mean', suite.Mean()                                     Kod 32
print 'Median', PMF.Percentile(suite,50)
```

Mean iznosi 55.95, dok median iznosi 56.

Zatim je dobro izračunati i interval kredibiliteta (*Kod 33).*

```
print'CI', PMF.CredibleInterval(suite,90)              Kod 33
```

Rezultat iznosi 51,161.

Problem je u tome što se želi saznati da li novčić balansiran i pošten. Pošto kredibilitet intervala ne iznosi 50%, ovo govori da novčić nije pošten. Naime, pitanje na početku je bilo vezano za to da li svi navedeni podaci podržavaju hipotezu da je novčić pristran više nego pošten. Da bi se pravilno moglo odgovoriti na ovo pitanje, moralo bi se točno definirati što se misli pod time da se podaci koriste kao dokaz za hipotezu. O ovome će biti više riječi u kasnijim dijelovima. Na početku se koristio uniformirani prior, što možda i nije bila najbolja ideja. Ako se uzme u obzir da novčić može biti deformiran kroz korištenje, onda vrijednost varijable x neće biti ni blizu 50%. Barem se može zaključiti da je nemoguće da su uzeli novčić koji je toliko deformiran i razbijen da mu je vrijednost varijable x manje od 10% ili veća od 90% [1] [6]. Bilo bi puno bolje da se odabere prior koji daje veću vjerojatnost da je x blizu 50%, a manju vjerojatnost da će x biti nekakva ekstremna vrijednost. Čak i sa totalno različitim priorima, distribucije posteriora su jako slične. Srednje vrijednosti i interval kredibiliteta su identični, što znači da se razlikuju manje od 0.5%. Ovo je zapravo bio primjer razmjene priora. Sa dovoljno podataka, ako se koristi potpuno različite priore, svi će početi konvergirati na isti posterior [1] [6] .

4.1. Optimizacija postupka

Do sada se prikazalo rješavanje problematike i primjena analize kroz kratke zadatke koji su se izvršavali relativno brzo. U slučaju se same metode budu morale koristiti na većim skupom podatak, potrebno je cijeli proces ubrzati i optimizirati. Prvu priliku za optimiziranje se smanjivanjem normaliziranja objekta suite. Prije se pozivalo metodu za osvježavanje jednom za svaki put kada bi zavrtjeli kovanicu (*Kod 34).*

```
dataset = 'G' * glava + 'R' * rep                                    Kod 34
for data in dataset:
      suite.Update(data)
```

Sama metoda za osvježavanje je izgledala ovako:

```
def Update(self,data):                                               Kod 35
      for hypo in self.Values():
            like = self.Likelihood(dana, hypo)
            self.Mult(hypo, like)
```

Svako pozivanje metode za osvježavanje prolazi kroz sve hipoteze, a zatim poziva metodu za normaliziranje podataka koja također ponovno prolazi kroz sve hipoteze (*Kod 35).*

Suite omogućava korištenje metode UpdateSet koja će olakšati ovaj posao (*Kod 36).*

Kod 36

```
def UpdateSet(self, dataset):
      for data in dataset:
            for hypo in self.Values():
                  like = self.Likelihood(dana, hypo)
                  self.Mult(hypo, like)
            return self.Normalize()
```

Zatim je potrebno pozvati metodu za osvježavanje na sljedeći način.

Kod 37

```
dataset = 'G' * glava + 'R' * rep
suite.UpdateSet(dataset)
```

Ovakva optimizacija iznimno ubrzava postupak, ali brzina izvršavanja i dalje ovisi o samoj količini podataka s kojima se barata. Stvari se mogu još više ubrzati ako se promijeni metodu za izračunavanje ishoda na način da se izvršava nad cijelim setom podataka, a ne jedan po jedan.

Alternativno, moguće je promijeniti cijeli paket podataka (engl. Dataset) u objekt tuple koji će sadržavati dva broja. Jedan broj će označavati broj glava dok će drugi označavati broj repova (*Kod 38).*

Kod 38

```
def Likelihood(self, data, hypo):
      x = hypo / 100.0
      if data == 'G':
            return x
      else:
            return 1 - x
```

Zatim se ponovno pozove metodu osvježavanja (*Kod 39).*

```
heads, tails = 140, 110                                    Kod 39
suite.Update((heads,tails))
```

4.2. Beta distribucija

Do sada se koristilo samo objekte klase Pmf da bi predstavilo diskretni set podataka varijable X. Ako je sada potrebno koristiti neprekidnu distribuciju, koristiti će se beta distribuciju.

Beta distribucija je neprekidna distribucija vjerojatnosti definirana na intervalu između nule i jedinice, gdje su nula i jedinica također uključeni u interval. Što je idealno jer zapravo pomoću toga se mogu predstaviti sve vjerojatnosti. Nadalje, sama beta distribucija funkcionira na isti način kao i Bayes-ov teorem na način da koristi istu distribuciju za prior i posterior kao i beta distribucija. Sami oblik beta distribucije ovisi o dva parametra, a to su alfa i beta. Ako je prior jednak beta distribuciji sa parametrima alfa i beta, razlikujemo podatak da postoji N broj puta kada je pala glava i M broj puta kada je pao rep, posterior će zbog ovoga biti beta distribucija sa parametrima alfa + M i beta + N [9].

Ovo će koristiti jedino ako je moguće pronaći beta distribuciju dovoljno dobru da je se može iskoristiti kao prior. Srećom, za popriličan broj priora postoji beta distribucija koji su poprilično precizni. Beta distribucija za alfa = 1 i beta = 1 je uniformna od nule do jedinice.

Da bi se ovo iskoristilo potrebno je koristiti klasu koja će predstavljati beta distribuciju (*Kod 40).*

```
class Beta(object):                                        Kod 40
      def __init__(self, alpha = 1, beta = 1):
            self.alpha = alpha
            self.beta = beta
```

Ovako se osiguralo da će distribucija biti uniformna.

Zatim još jednom se poziva metodu za osvježavanje (*Kod 41).*

```
def Update(self, data):
      glave, repovi = data
      self.alpha += glave
      self.beta += repovi
```

Kod 41

Data sadržava broj glava i repova.

Koristeći beta distribuciju, zapravo dobijemo alternativni način za riješiti Euro problem.

```
beta = new Beta()
beta.Update((140,110))
print beta.Mean()
```

Kod 42

Ovdje se ponovo dobije srednju vrijednost koja iznosi ni manje ni više nego 56% (*Kod 42).*

Kroz beta distribuciju je također moguće koristiti i metodu EvalPDF, koja je zapravo funkcija vjerojatnosti gustoće (PDF).Ova metoda omogućava da se stvori diskretna aproksimacija same distribucije. U teoriji vjerojatnost, **PDF (funkcija vjerojatnosti gustoće)** (engl. Probability density function) odnosno gustoća neprekidne nasumične varijable, je funkcija koja opisuje relativan ishod da će nasumična varijabla dobiti nekakvu vrijednost. Vjerojatnost da će nasumična varijabla pasti u određen raspon se izračunava kroz integral gustoće samog raspona. Bez predubokog ulaska u teoriju o ovoj funkciji, dovoljno je znati da pomoću nje možemo izračunavati aproksimaciju raznih distribucija i vjerojatnost podataka nad danim intervalima [10].

U ovom poglavlju je prikazan način rješavanja problema sa različitim priorima, pod uvjetom da tijekom procesa se dođe do što više pozadinskih podataka, zapravo svi konvergiraju na isti rezultat. Iznimno je bitno kod Bayes-ove analize da u slučaju ako se ne odabire dva ista modela, interpretacija podataka se može bitno razlikovati. Tako da zapravo kod istih podataka, može doći do totalno različitih ishoda i posteriora. Na primjer, ako se pretpostavi da je vrijednost varijable X kod Euro problema manji od 5%, i ako pridoda vjerojatnost 0 na sve ostale hipoteze, nikakvi podaci ga neće razuvjeriti u tome. Kod Bayes-ovih analitičara, popularna je priča o Cromwellu i njegovom pismu glavnoj Crkvi u Škotskoj.

„*I beseech you, in the bowels of Christ, think it possible that you may be mistaken.“*

Što zapravo, kroz jednu šaljivu analogiju govori da bi se trebalo izbjegavati pridodavanje vrijednosti 0 na niti jednu hipotezu prije ikakvih izračuna [1], [6], [10].

5. Zaključak

Prolazeći kroz osnovne i jednostavnije primjene Bayes-ove analize i teorema, pa sve do kompliciranijih primjena prikazano je koliko su ove metode ključne za čitavu granu statistike. Na samome kraju se samo malo dotaknulo najnaprednijih oblika i najkompliciranijih zadataka koji se danas koriste za mnoge izračune i donošenje mnoštva odluka od iznimne važnosti za razne industrije. Bayes se danas koristi i u istraživanju mnogih različitih bolesti, a pogotovo i u polju genetike. Korištenjem ovih modela vjerojatnosti moguće je pomoći velikom broju populacije u ranoj prevenciji određenih bolesti i mogućih zdravstvenih problema. Ovakve analize i modeli se čak koriste i u istraživanju raznih životinjskih vrsta i mogućnosti njihovog izumiranja, dok je pogotovo zanimljivo i istraživanje bakterija koji se nalaze u ljudskom tijelu.

Kroz ovaj pregled analize i teorema se prošlo kroz razne faze. Na početku su prezentirani jednostavni primjeri Bayes-a i njegove filozofije. Sama osnova njegove analize je ideja da se koriste distribucije vjerojatnosti da bi se predstavilo nekakve mogućnosti i vjerovanja koja na prvu se čine nemoguća. Kasnijim primjerima je pokazano kroz razne primjene kako se može bez raznih matematičkih formula, zapravo stvoriti jedan okvir za rješavanje problema. Okvir koji možemo gledati kao kocke za građu koje možemo aranžirati kako želimo da bi se riješilo stvarne životne probleme. Za većinu pravih problema koriste se snažne simulacije modela koje pokušavaju riješiti određene dijelove. Teško je kroz simulacije uračunati sve potencijalne faktore i elemente koji se mogu pojaviti u raznim situacijama, ali zato nam ova analiza omogućuje da ponovnim iteracijama i modulacijom stvaramo kompleksnije modele koji će nam pomoći doći do pravih rezultata.

6. Literatura

[1] Think Bayes – Allen B.Downey 2013.

[2] Bayesov Teorem

https://en.wikipedia.org/wiki/Bayes'_theorem

[Pristupljeno prosinac 2016. godine]

[3] Bayes primjeri

https://betterexplained.com/articles/an-intuitive-and-short-explanation-of-bayes-theorem/

[Pristupljeno prosinac 2016. godine]

[4] Bayes primjeri

http://allendowney.blogspot.hr/2011/10/my-favorite-bayess-theorem-problems.html

[Pristupljeno prosinac 2016. godine]

[5] Bayes primjeri

http://allendowney.blogspot.hr/2011/10/all-your-bayes-are-belong-to-us.html

[Pristupljeno prosinac 2016. godine]

[6] Bayes primjeri

http://www.markhneedham.com/blog/2015/05/31/r-think-bayes-euro-problem/

[Pristupljeno prosinac 2016. godine]

[7] Bayes primjeri

http://allendowney.blogspot.hr/2011/04/bayesianness-is-next-to-godliness.html

[Pristupljeno prosinac 2016. godine]

[8] Bayesov Teorem ~ *Green Tea Press*

http://www.greenteapress.com/thinkbayes/html/thinkbayes002.html

[Pristupljeno prosinac 2016. godine]

[9] Beta distribucija

https://en.wikipedia.org/wiki/Beta_distribution

[Pristupljeno prosinac 2016. godine]

[10] Masovna funkcija vjerojatnosti

https://en.wikipedia.org/wiki/Probability_mass_function

[Pristupljeno prosinac 2016. godine]

[11] Uvjetna vjerojatnost

https://en.wikipedia.org/wiki/Conditional_probability

[Pristupljeno prosinac 2016. godine]

[12] Zajednička vjerojatnost

https://en.wikipedia.org/wiki/Conjunction_fallacy

[Pristupljeno prosinac 2016. godine]

[13] Zakon moći

https://en.wikipedia.org/wiki/Power_law

[Pristupljeno prosinac 2016. godine]

7. Sažetak

Kada se pokušava objasniti nešto što ima kompleksnu i opširnu primjenu kao što to ima Bayes-ov teorem, jako je teško zapravo predstaviti ga u punom svjetlu. Ovaj teorem, iako jednostavan u teoriji, ima neizmjerno veliku i kompliciranu primjenu kroz razne industrijske, statističarske i gospodarske grane. Cilj je kroz ovaj rad napraviti kratak uvod i pregled osnova Bayes-ovog teorema i njegove primjene. Počevši od definiranja teorema i njegovih osnovnih karakteristika, prelazimo na pregled teorema kroz najjednostavnije probleme. Počevši od jednostavnih problema kao što je problem izvlačenja kolačića iz nasumičnih zdjela, Elvis-ovog brata blizanca i drugih problema, možemo objasniti na jednostavnim primjerima osnove Bayes-ove statistike i njegovog teorema. Kroz rad prijeći ćemo i na kompliciranije probleme gdje ćemo se prikazati kako korištenje cijele teorije vjerojatnost i Bayes-ovog teorema ima primjenu u stvarnom životu i kompliciranijim problemima. Problemi kao što su Oliver-ova krv, njemački tenkovi i Monty Hall su samo neki od primjera stvarnih i čestih problema koji se mogu riješiti koristeći teoriju vjerojatnosti. Nemoguće je prikazati potpune mogućnosti Bayes-ove statistike i njegovog teorema, stoga ćemo prikazati kratak pregled i uvod u napredne mogućnosti samog teorema u rješavanju višedimenzionalne problematike. Oni koji žele saznati više o naprednim mogućnostima i raznim primjenama, mogu pronaći razne reference i literature u posebnom poglavlju.

Ključne riječi - Bayes, statistika, Bayesov teorem, Python, vjerojatnost, pretpostavke, procjene, Euro problem, napredne primjene, distribucije podataka vjerojatnosti

8. Summary

When trying to explain something as complex as the Bayes theorem, it's rather hard to do it justice. This theorem, even though it is simple in theory, has a rather huge and complex application throughout different industrial, statisthical, economical and business enterprises and institutions. It's hard to really shine a light on the full spectrum of it's possibilities. However, the goal was to make a short and intuitive compedium of the basics of the Bayes theorem and it's application. Starting off with the definition of the theorem and it's basic characteristics, we advance to the implementation of the theorem in the simplest of problems. Starting with the simple problems, such as pulling a vanilla cookie from random bowls, the identical twin of Elvis and others , we try to explain the basics of the Bayesian statistics and it's theorem. Throughout the compedium, we shall go over the more complicated problems where we shall display how using the entire theory of probability and the Bayesian theorem has a significant role in every day tasks and more complex issues. Problems such as Oliver's blood, german tanks and Monty Hall, are just some of the often and real issues that can be solved using the theory of probability. Since it is impossible to showcase the full potential of the Bayesian statistics and it's theorem we will showcase a short overview into the more advanced and really complex capabilities and applications of the theorem when dealing with multidimensional problems. Those who want to know more, can find references and literature in the appropriate section.

Key words - Bayes, statistika, Bayesian theorem, Python, probability, assumptions, evaluations, Euro problem, advanced applications, data distribution , likelihood