Kolegij: Otkrivanje znanja u skupovima podataka

Aplikacija za dubinsku analizu podataka potrošačkih košarica

Seminarski rad

Zagreb, 2007.

SADRŽAJ

1. UVOD

Analiza pravila pridruživanja unutar potrošačkih košarica je poseban primjer dubinske analize podataka gdje je cilj otkriti koji se artikli (proizvodi) zajedno pojavljuju unutar potrošačkih košarica. Osobit zadatak ovog tipa analize je otkriti kombinacije artikala koji se zajedno pojavljuju u potrošačkim košaricama a da pri tom njihovo zajedničko pojavljivanje nije unaprijed očekivano (npr. zajedničko pojavljivanje kruha i mlijeka u jednoj potrošačkoj košarici nije osobito informativno). Analiza pravila pridruživanja osobito je interesantna jer rezultati analize omogućavaju predviđanje što će kupac uraditi (ili bolje reći kupiti) ako je već u svoju košaricu stavio određene artikle. Pravilo pridruživanja u običnom jeziku možemo iskazati na sljedeći način (primjer je naveden simbolično):

„Ako je kupac kupio 'Coca-Colu' i 'Pjenu za brijanje' onda je 20 puta veća šansa da će kupiti i 'sredstvo za održavanje kontaktnih leća'“

Poznavanjem pravila pridruživanja u potrošačkim košaricama dobiva se bolji uvid u potrošačke navike kupaca što se onda iskorištava za pokretanje ciljanih reklamnih kampanja, za optimiranje cijena unutar neke grupe proizvoda, za optimalno razmještanje proizvoda po policama dućana, za oblikovanje posebnih prodajnih ponuda u maloprodajnim lancima, za oblikovanje prodajnih kataloga itd.

2. PRAVILA PRIDRUŽIVANJA

2.1 OSNOVNI POJMOVI

Kod analiziranja pravila pridruživanja cilj je otkriti zanimljive uzorke ili pravila unutar analiziranog skupa podataka. U hrvatskoj stručnoj terminologiji ovaj tip dubinske analize podataka naziva se otkrivanje pravila pridruživanja ili asocijacijska pravila (engl. *Association Rules*). Primjeri ovakve analize su otkrivanje potrošačkih navika kupaca u maloprodaji (*market basket analysis*), otkrivanje koji niz transakcija sugerira na prevare u kartičnom poslovanju, analiza navika posjetitelja web portala s ciljem bolje personalizacije istog prema određenom tipu posjetitelja itd. Kod ovakvih analiza osobito je važno domensko znanje osobe ili tima koji obavlja analizu jer je bitno uočiti one uzorke i pravila koji su interesantni i sami po sebi nisu očiti. Postoji više algoritama koji se koriste za ovu svrhu a najpoznatiji je ***apriori*** algoritam [agr94].

U terminologiji asocijacijskih pravila susrećemo se sa sljedećim terminima:

- Element ili dio (*item*).
- Skup elemenata (*itemset*).
- Transakcija (*transaction*).
- Značaj (*support*).
- Pouzdanost (*confidence*).

Element predstavlja osnovni (sastavni) dio transakcije. Transakcije se sastoje od skupa elemenata - *itemsetova*, i međusobno se razlikuju po broju elemenata te rednim brojem transakcije. Svaka transakcija u skupu transakcija daje nam informaciju o tome koji elementi se zajedno pojavljuju u transakcijama. Korištenjem transakcija moguće je napraviti tablice koje nam daju frekvencije pojavljivanja parova određenih elemenata u transakcijama.

Općenito pravilo ili uzorak se prikazuje kao niz konjunkcija i disjunkcija sastavljenih od elemenata. Pojmove **značaj** i **pouzdanost** pravila objašnjava sljedeći primjer:

Neka je R_1 asocijacijsko pravilo koje glasi:

R_1="*Element A **pojavljuje** se zajedno sa elementom B u 10% svih transakcija*" (1)

Vrijednost 10% predstavlja mjeru frekvencije pojavljivanja para elemenata *A-B u* skupu svih transakcija i predstavlja **značaj** (*support*) para elemenata, te ga označavamo sa *supp*(AB)=10%.

Općenito vrijedi:

$$\text{supp}(A) = \frac{\text{br. trans. s elem. A}}{\text{ukupan broj trans.}} * 100\% \qquad (2)$$

Prilikom određivanja asocijacijskih pravila potrebno je prethodno definirati tzv. "minimalni značaj" (*minimum support*), koji predstavlja najmanju dozvoljenu frekvenciju pojavljivanja, i svaki element koji čini asocijacijsko pravilo mora zadovoljiti uvjet da je njegov značaj veći ili jednak tom minimalnom.

Ako je frekvencija pojavljivanja elementa A u svim transakcijama, *supp*(A)=15%, a elementa B *supp*(B)=10%, tada omjer broja transakcija u kojima se pojavljuju oba elementa i broja transakcija u kojima se pojavljuje element A (koji predstavlja uvjetni dio pravila) nazivamo **pouzdanost** (*confidence*) pravila i označavamo sa *conf* (R_1). Vrijedi:

$$conf(R_1) = \frac{\text{supp(AB)}}{\text{supp(A)}} * 100\% \qquad (3)$$

U našem navedenom primjeru, pouzdanost pravila R_1 od 0.666 jednaka je tvrdnji da, kada se u transakciji pojavi element A, postoji 66.6% vjerojatnosti da će se u istoj transakciji pojaviti i element B.

Za asocijacijska pravila važno je unaprijed definirati iznos minimalnog značaja i pouzdanosti tako da pravila čija je pouzdanost i značaj manji od minimalnih vrijednosti ne uzimamo u obzir. Definiranjem minimalnog značaja i pouzdanosti pravila smanjuje se vrijeme pronalaženja pravila jer je pretraživanje cijelog prostora mogućih pravila za dani skup elemenata kombinatorički problem gdje je mogući broj pravila eksponencijalno ovisan o broju različitih elemenata. Zbog toga za veći broj elemenata pretraživanje cijelog prostora mogućih pravila bez navedenih ograda postaje neizvedivo u realnom vremenu. Iznosi minimalnog značaja i pouzdanosti pravila koji će se uzimati u obzir određuju se eksperimentalno ili po iskustvu analitičara ovisno o konkretnoj situaciji u kojoj se vrši analiza.

Ipak, ako je cilj pronaći neki rijetki uzorak (npr. onaj koji sugerira na neki kvar ili na zloupotrebu kreditne kartice) koji se recimo pojavljuje jednom u 100000 slučajeva ili rjeđe, onda se mora pretraživati cijeli prostor mogućih rješenja što može trajati jako dugo ovisno o broju različitih elemenata koji se mogu pojaviti u transakcijama.

Važno je spomenuti da se značaj prije svega odnosi na element ili skup elemenata (*itemset*), dok se pouzdanost odnosi na pravilo (obrnuti pojmovi ne vrijede, dakle ne možemo govoriti o pouzdanosti elementa i o značaju pravila).

2.1.1 Formalna definicija problema

Metodu asocijacijskih pravila mogli bismo formalno definirati na sljedeći način:

Neka je **I** = {I_1, I_2, …, I_n} skup svih elemenata koji se nalaze u nekoj bazi podataka i **D** skup svih transakcija **T**, koji može biti prikazan u obliku tablice spremljene u relacijskoj bazi podataka, dimenzijske tablice ili u obliku obične podatkovne datoteke.

Svaka pojedina transakcija predstavlja skup elemenata, takvih da vrijedi da je **T** podskup od **I** ($T \subseteq I$). Svaka transakcija **T** ima svoj jedinstveni identifikator TID. Kažemo da transakcija **T** sadrži **X** (skup elemenata iz **I**) ako vrijedi da je **X** podskup od T ($X \subseteq T$).

Asocijacijsko pravilo pišemo u obliku $\mathbf{X \rightarrow Y}$ gdje je X skup elemenata koji se nalazi na desnoj strani pravila a Y skup elemenata na lijevoj strani pravila i za koje vrijedi:

$$X \subset I, Y \subset I, X \cap Y = 0 \qquad (4)$$

Ovo pravilo ima pouzdanost ***c***, ako c% transakcija koje sadrže X također sadrže i Y, te značaj ***s*** u skupu transakcija, ako s% svih transakcija u **D** sadrži i X i Y ($X \cup Y$).

Problem pronalaženja asocijacijskih pravila u skupu transakcija **D**, svodi se dakle na pronalaženje svih asocijacijskih pravila koja imaju pouzdanost veću od neke minimalne pouzdanosti, odnosno da je značaj elemenata koji čine to pravilo veće od minimalnog značaja koje se unaprijed odredi.

Za pravila vrijedi:

- Pravila $X \rightarrow Y$ i $Y \rightarrow X$ predstavljaju dva različita pravila sa različitim iznosima pouzdanosti. Razlog tomu je što općenito govoreći skupovi X i Y imaju različit značaj te stoga po definiciji (3) navedena pravila imaju različitu pouzdanost.
- Postojanje pravila $X \rightarrow A$ ne mora nužno značiti da vrijedi pravilo $X+Y \rightarrow A$ jer se u ovom slučaju može dogoditi da je ukupni značaj *supp*(X+Y) manji od predefiniranog minimalnog značaja.
- Postojanje pravila $X \rightarrow Y$ i $Y \rightarrow Z$ ne mora nužno značiti da postoji pravilo $X \rightarrow Z$ jer pouzdanost pravila $X \rightarrow Z$ može biti manja od minimalne pouzdanosti.

Iako općenito broj elemenata na desnoj strani pravila pridruživanja Y može biti i veći od 1 u praktičnim implementacijama aplikacija za pronalaženje pravila pridruživanja broj elemenata na desnoj strani je redovito jedan.

2.1.2 Otkrivanje pravila (apriori algoritam)

Problem otkrivanja svih asocijacijskih pravila mogli bismo podijeliti na tri dijela:

- Ograničenje broja elemenata.
- Otkrivanje velikog (pogodnog) skupa elemenata.
- Stvaranje pravila.

Kada stvaramo asocijacijska pravila, mi ustvari tražimo **povezanost** među skupovima elemenata. Broj kombinacija za skupove sa više elemenata raste približno eksponencijalno s brojem elemenata u transakcijama, tako da broj mogućih asocijacija

jako brzo raste s povećanjem broja različitih elemenata. Tako, na primjer, za 1000 različitih produkata u nekoj trgovini, ukupan broj mogućih skupova od tri elementa je:

$$\binom{n}{k} = \binom{1000}{3} = 166.167 * 10^6 \quad !! \tag{5}$$

Iz ovog je vidljivo da računanje značaja ili drugih mjera za skupove elemenata s pet ili više elemenata postaje vremenski neizvedivo bez obzira kako "inteligentan" algoritam imali. Stoga je važno koristiti tzv. taksonomiju, odnosno kategorizaciju elemenata. Izbor pravog nivoa kategorizacije može igrati ključnu ulogu u smislenosti konačnih pravila, ali i redukciji velikog broja artikala u jedan element. Deseci, ponekad i stotine artikala mogu biti svedeni na jednu ili više kategorija koje dobro reprezentiraju generalna svojstva tih artikala.

Da bi se izbjeglo pretraživanje cijelog prostora mogućih pravila, koji je najčešće toliki da ga je nemoguće pretražiti u konačnom vremenu, traže se samo pravila sastavljena od skupova elemenata *(itemsetova*) koja imaju značaj veći od unaprijed određenog minimuma. Takvi skupovi nazivaju se **veliki** skupovi elemenata (*large itemsets* ili *frequent itemsets* [han01]), a svi ostali skupovi nazivaju se mali. Neki od najpoznatijih algoritama, kojima pronalazimo asocijativna pravila, otkrivaju velike skupove elemenata na način višestrukog prolaska kroz bazu podataka.

Najčešće korišteni način pronalaska velikog skupa elemenata temelji se na tzv. **apriori**-triku, koji govori da je: *svaki podskup velikog skupa elemenata također velik*, tj. ako je skup **S** velik onda je i svaki njegov podskup velik. Da bismo pronašli sve velike skupove podataka (*itemsetove*), prvim prolaskom kroz bazu podataka brojimo frekvenciju pojavljivanja svakog pojedinog elementa u bazi, tj. računamo značaj svakog elementa. One elemente čiji je značaj veći od minimalnog značaja proglasimo velikim elementima, i ti veliki elementi čine skup velikih elemenata (1-*itemset*). U svim sljedećim koracima promatramo samo velike *itemsetove* (generirane u prethodnom prolazu), i njih koristimo za stvaranje novih, potencijalno velikih skupova elemenata, koje nazivamo kandidatima. Skup kandidata koji ima **k+1** elemenata, generiran je od *itemsetova* veličine **k** (tzv. *k-itemset*). Na kraju pojedinog prolaza, računamo značaj svakog kandidata iz skupa kandidata, svi oni kandidati čiji je značaj veći od minimalnog postaju elementi u *(k+1)-itemsetu.*

Ovaj proces se ponavlja sve dok se više ne može pronaći novi veliki *itemset* (za jedan element veći od prethodnog).

Drugi način pronalaska velikih *itemsetova* temelji se na principu "podijeli pa vladaj". Naime, neku bazu podataka možemo podijeliti na više dijelova - particija (ne nužno iste veličine). U svakoj od tih particija pronađemo velike skupove elemenata. Skup elemenata (*itemset*) može biti velik samo ako je velik u najmanje jednoj od ovih particija.

Za stvaranje pravila koristimo velike skupove elemenata (*itemsetove*) koje određujemo na definirani način. Naime, za svaki veliki *itemset* **B**, pronađemo sve ne-prazne podskupove od **B** (zbog apriori-trika, svaki takav podskup bit će također velik). Za svaki takav podskup **A** stvorimo pravilo:

$$A \rightarrow (B - A) \tag{6}$$

ako ono ima pouzdanost veću od unaprijed definirane minimalne pouzdanosti, tj. ako vrijedi:

$$conf\,(A) = \frac{\text{supp(B)}}{\text{supp(A)}} \geq \text{minconf} \tag{7}$$

2.1.3 Apriori algoritam

Apriori algoritam prvi put je opisan u [agr94] i koristi se za pronalaženje pravila pridruživanja analiziranjem transakcija. Apriori algoritam se temelji na apriori triku i moguće su različite implementacije tog algoritma a na ovom mjestu će biti opisana jedna od tih implementacija.

Prvim prolaskom kroz bazu, prebrojava se svaka pojava pojedinog elementa, kako bismo otkrili sve velike 1-*itemsetove* $\mathbf{L_1}$ (*frequent itemsets* [han01]). U drugoj iteraciji *parovi* elemenata iz skupa $\mathbf{L_1}$, postaju kandidati, točnije elementi skupa kandidata $\mathbf{C_2}$. Prolazeći kroz bazu, utvrđujemo značaj svakog elementa iz $\mathbf{C_2}$, svi oni elementi čiji je značaj veći od minimalnog postaju elementi skupa $\mathbf{L_2}$. Tako se za svaki sljedeći korak, npr. korak ***k***, može reći da se sastoji od dvije faze:

- U prvoj fazi se veliki *itemset* $\mathbf{L_{k-1}}$ (koji je pronađen u **k-1** koraku) koristi za stvaranje skupa kandidata $\mathbf{C_k}$, koristeći **apriori-gen** funkciju.
- U drugoj fazi se utvrđuje značaj svakog pojedinog kandidata iz skupa kandidata $\mathbf{C_k}$. Oni kandidati koji imaju značaj veći od minimalnog, i čiji se svi podskupovi nalaze u $\mathbf{L_{k-1}}$, postaju elementi skupa $\mathbf{L_k}$.

Apriori algoritam izvodimo u onoliko koraka koliko nam je potrebno (a to je određeno s parametrom koji definira maksimalnu duljinu pravila **K**), ili dok *itemset* $\mathbf{L_k}$ ne postane prazan skup. Na kraju kad su pronađeni svi veliki *itemsetovi* od njih se formiraju pravila i to tako da se u obzir uzimaju samo pravila čija je pouzdanost veća od minimalne.

Logički se *apriori* algoritam može napisati u pseudokodu:

```
1)  L1  = {large 1-itemsets};
2)  for ( k = 2; Lk-1 ≠ 0; k++ ) do begin
3)     Ck  = apriori-gen(Lk-1); // New candidates
4)     forall transactions t ∈ D do begin //DB
5)            Ct  = subset(Ck, t); // Candidates contained in t
6)                   forall candidates c ∈ Ct  do
```

```
7)                          c.count++;
8)           end
9)  Lk = {c ∈ Ck  | c.count ≥ minsup}
10) end
11) Answer = ⋃k Lk;
```

L_k - predstavlja skup velikih *k-itemsetova,* **C_k** - predstavlja skup kandidata *k-itemset*.

Funkcija *apriori-gen* služi za otkrivanje kandidata. Njen argument je **L_{k-1},** skup svih velikih *(k-1)-itemsetova,* a kao rezultat dobijemo skup kandidata **C_k**, veličine **k**. Funkcija se sastoji od dva koraka: "*join step*" (korak ujedinjavanja) i "*prune step*" (korak pročišćavanja).

U prvom koraku spajamo elemente skupa **L_{k-1}** s ciljem dobivanja skupa kandidata **C_k.** Ovaj korak se može opisati kako pronalaženje svih parova {**U**,**V**} gdje su **U** i **V** iz **L_{k-1}** takvih da je njihova unija ima duljinu **k**. Kompleksnost ovog koraka je u najgorem slučaju O(| **L_{k-1}** |^3) gdje je | **L_{k-1}** | broj velikih *itemsetova* veličine k-1. No u praksi ta složenost je najčešće linearna s obzirom na broj velikih *itemsetova* u svakom koraku.

U drugom koraku pročišćavamo skup kandidata **C_k** na način da brišemo sve one kandidate **c** iz **C_k** čiji se neki od podskupa veličine *(k-1)* ne nalazi u **L_{k-1}**.

Primjer za apriori-gen funkciju:

Neka je L_3 = {{123},{124},{134},{135},{234}}

Nakon "*join step*" skup kandidata je C_4 = {{1234},{1345}}.

U slijedećem koraku dobivamo za c_1={1234}, skup njegovih (k-1) podskupova s_1={{123},{124},{134},{234}} a za c_2={1345}, skup njegovih (k-1) podskupova je s_2={{134},{135},{145},{345}}. Kako se skupovi {345} i {145} ne nalaze u L_3, c_2 ćemo izbaciti iz C_4 i kao konačno rješenje *apriori-gen* funkcije dobit ćemo C_4= {{1234}}.

Zadnji korak algoritma, kad su pronađeni je da se u obzir uzmu samo ona pravila čija je pouzdanost veća od zadane vrijednosti.

Značaj kandidata iz skupa **C_k** se određuje tako što se prebroji koliko transakcija sadrži te kandidate. Značaj pojedinog kandidata možemo odrediti jednostavnim načinom*:* prolazeći kroz bazu kako bi za svaku transakciju utvrdili sadrži li ona taj kandidat , što s obzirom da se vrlo često radi o velikom broju transakcija i mogućih kandidata ovaj način čini jako sporim i stoga neupotrebljivim. Zbog toga se koriste razni trikovi koji ubrzavaju pronalaženje kandidata koji imaju značaj veći od zadanog primjerice organiziranjem kandidata u strukture slične stablima koji se onda brže mogu pretraživati.

Vremenska složenost *apriori* algoritma dana je sljedećim izrazom [han01]:

$$O(\sum_{k}^{K}|C_k|np) \qquad (8)$$

Gdje je $|C_k|$ broj kandidata duljine **k**, **K** maksimalna duljina pravila, **n** ukupni broj transakcija i **p** broj različitih elemenata koji se pojavljuju u transakcijama. Iz izraza 8 je očito da se na vremensko izvođenje algoritma može utjecati na više načina:

- Ograničavanje broja kandidata u svakoj od iteracija algoritma – To se postiže podešavanjem parametra Supp (minimalni značaj pravila). Ako se taj parametar postavi premali onda se izlažemo opasnosti da broj kandidata C_k u svakoj iteraciji algoritma bude preveliki tako da algoritam ne bude izvodljiv u realnom vremenu, a ako se postavi odveć veliki onda se može dogoditi da se ne pronađe niti jedno pravilo koje zadovoljava postavljeni uvjet. Zbog toga je kod implementacije aplikacija koje koriste apriori algoritam za pronalaženje pravila pridruživanja dobro odrediti granice u kojima se može kretati minimalni iznos značaja. Te granice se određuju eksperimentalno a uvjetovane su brojem različitih elemenata koji se pojavljuju u transakcijama te brojem transakcija.
- Ograničavanje maksimalne duljine pravila **K** – U praksi nas obično interesiraju pravila manjih duljina (obično ne više od 5). Pravila s većim brojem elemenata obično imaju zanemarive značaje u odnosu na pravila s manjim brojem elemenata.
- Smanjivanje broja transakcija koje se analiziraju **n** – Ovo se postiže uzimanjem reprezentativnog uzorka iz cijelog skupa podataka. Tom prilikom treba biti oprezan i stvarno se osigurati da je uzorak reprezentativan inače se dobiveni rezultati ne mogu odnositi na cijelu populaciju.
- Smanjivanje broja različitih elemenata **p** – Prema jednakosti (8) ovaj parametar utječe na vrijeme izvođenja algoritma eksplicitno i implicitno tako što utječe na broj kandidata za velike *itemsetove* u svakoj iteraciji algoritma. Na njega se može utjecati uvođenjem odgovarajuće taksonomije pri čemu se elementi mogu grupirati u više grupa (ili klasa) prema svojim svojstvima. Odabiranje dobre taksonomije može imati drastičan utjecaj na skraćivanje vremena izvođenja algoritma a uz istodobno zadržavanje osnovnih zakonitosti zbog kojih se pravila pridruživanja uopće i analiziraju. Primjerice, informacija o kupovini piva i čipsa zajedno iz pravila (“Franck Čips” $\rightarrow$ “Ožujsko pivo”) u pravilu (“Čips” $\rightarrow$ “Pivo”) je sačuvana. Naravno dio informacije o vrsti čipsa i piva koji se kupuju zajedno je ovim putem izgubljen a to je cijena koja se plaća za ubrzavanje izvođenja algoritma. Ukoliko se načini dobra taksonomija mogu se birati nivoi na kojima se vrši analiza podataka ovisno o nekom specifičnom interesu analitičara.

2.1.4 Interesantnost pravila

Ovisno o karakteristikama podataka koji se analiziraju čak i uz uvedena ograničenja na pouzdanost i značaj pravila, dobiveni broj pravila može biti jako veliki, reda veličine 10^5 pa i više. Očito, u tom slučaju pronalaženje najinteresantnijih pravila može biti jako težak zadatak. Ako se ne definiraju dodatni kriteriji za ocjenjivanje interesantnosti pravila

pridruživanja moguće je da ona najvrjednija pravila nikad ne budu uočena u velikom broju pronađenih pravila odlučivanja. Zbog toga je pored značaja i pouzdanosti kao osnovnih veličina za ocjenjivanje vrijednosti nekog pravila korisno definirati i dodatne kriterije prema kojima možemo određivati interesantnost nekog pravila. Dodatni kriteriji za ocjenjivanje interesantnosti pravila odlučivanja mogu biti neovisni ili ovisni o domeni na koju se podaci odnose.

Kod domenski neovisnih kriterija interesantnosti pravila pridruživanja uglavnom se radi o raznim statističkim veličinama pomoću kojih ili ocjenjujemo informativnost nekog pravila pridruživanja ili kolika je prediktivna moć nekog pravila pridruživanja.

Veličina koja se često koristi za ocjenjivanje kvalitete pravila pridruživanja je ***lift*** ili **poboljšanje** [hbd03]. *Lift* nam govori koliko puta smo sigurniji da će kupac kupiti proizvod koji čini desnu stranu pravila znajući lijevu stranu pravila, u odnosu na slučaj kad ne znamo lijevu stranu pravila (slučajno pogađanje). Formalno, za neko pravilo pridruživanja $X \rightarrow Y$, lift se izračunava na sljedeći način:

$$Lift(X \rightarrow Y) = \frac{Conf(X \rightarrow Y)}{Supp(Y)} \tag{9}$$

Brojnik izraza (9) predstavlja pouzdanost cijelog pravila a nazivnik je značaj desne strane pravila koji u stvari predstavlja vjerojatnost da ćemo slučajnim odabirom odabrati proizvod Y ako nasumce odabiremo proizvode. Naravno, sve navedene veličine se izračunavaju iz podataka koji se analiziraju. Za već navedeno pravilo R_1 dano izrazom (1) *lift* iznosi:

$$Lift(R1) = \frac{Conf(A \rightarrow B)}{Supp(B)} = \frac{0.66}{0.1} = 6.6 \tag{10}$$

Lift (poboljšanje) za navedeno pravilo je 6.6 što znači da 6.6 puta imamo veću šansu da pogodimo da će kupac kupiti proizvod B ako već znamo da je kupio proizvod(e) A u odnosu na slučaj kad nam ta informacija nije poznata.

Lift kao veličina kojom procjenjujemo vrijednost pravila pridruživanja ima određeni nedostatak jer ne uzima u značaj cijelog pravila već samo njegovu pouzdanost.

Druga domenski neovisna veličina za ocjenjivanje interesantnosti pravila pridruživanja je veličina koja se u stručnoj literaturi naziva *J-measure* [han01]. Ova veličina se definira sljedećim izrazom:

$$J(X \rightarrow Y) = Supp(X)\left(Conf(X \rightarrow Y)\log\frac{Conf(X \rightarrow Y)}{Supp(Y)} - (1 - Conf(X \rightarrow Y)\log\frac{1 - Conf(X \rightarrow Y)}{1 - Supp(Y)}\right) \tag{11}$$

J-measure predstavlja informacijski dobitak (*cross-entropy*) o Y kada znamo lijevu stranu pravila X u odnosu na slučaj kad to ne znamo. Ova veličina daje ocjenu koliko se naše znanje o Y razlikuje u slučaju kad znamo X (tj. kad znamo da je kupac kupio proizvode predstavljene s X) u odnosu na slučaj kad nam ta informacija nije poznata, a sve uzevši u obzir i značaj pravila. Prema definiciji (9) *J-measure* uzima u obzir i značaj lijeve strane pravila. Zbog toga ova veličina daje bolju ocjenu statističke interesantnosti nekog pravila. Pravilo s većim iznosom *J-measure*-a je statistički gledano interesantnije.

Veličine opisane izrazima (9) i (11) ne uzimaju u obzir domenu iz koje dolaze podaci koji se analiziraju što im je u jednu ruku prednost a u drugu nedostatak. Prednost im je što se mogu koristi kao veličine za ocjenu interesantnosti pravila neovisno o problemu koji se analizira (analize potrošačkih košarica, analiza korištenja i uporabljivosti web portala, *fraud detection*, itd.). No, vrlo često domena problema koji se analizira može diktirati koja pravila smatramo interesantnijim od drugih, a pritom uzimajući u obzir osnovne veličine koje karakteriziraju neko pravilo pridruživanja. Tako bi primjerice kod analize potrošačke košarice bilo logično uzeti u obzir i vrijednost proizvoda koji se pojavljuju u pravilima pridruživanja kao faktor koji bi mogao utjecati na interesantnost pravila gledano sa stajališta menadžera prodaje u nekom trgovačkom lancu. Prema tome, pravila koja uključuju proizvode veće jedinične vrijednosti mogu dobiti prednost nad pravilima koja sadrže proizvode niže vrijednosti unatoč manjim iznosima značaja i pouzdanosti. Budući da su takvi kriteriji domenski ovisni, njihovo definiranje je heurističke prirode i njihova formalna definicija se mijenja od problema do probleme koji se analizira. U stručnoj literaturi predložen je veći broj domenski ovisnih kriterija interesantnosti pravila pridruživanja, ali niti jedan nije stekao širu uporabu baš stoga što su izrazito domenski ovisni.

Za potrebe analize pravila pridruživanja podataka trgovačkog lanca Konzum definiran je domenski ovisan kriterij interesantnosti pravila pridruživanja koji u obzir uzima i cijenu proizvoda koji čine neko pravilo odlučivanja. Formalna definicija ovog kriterija interesantnosti pravila pridruživanja je:

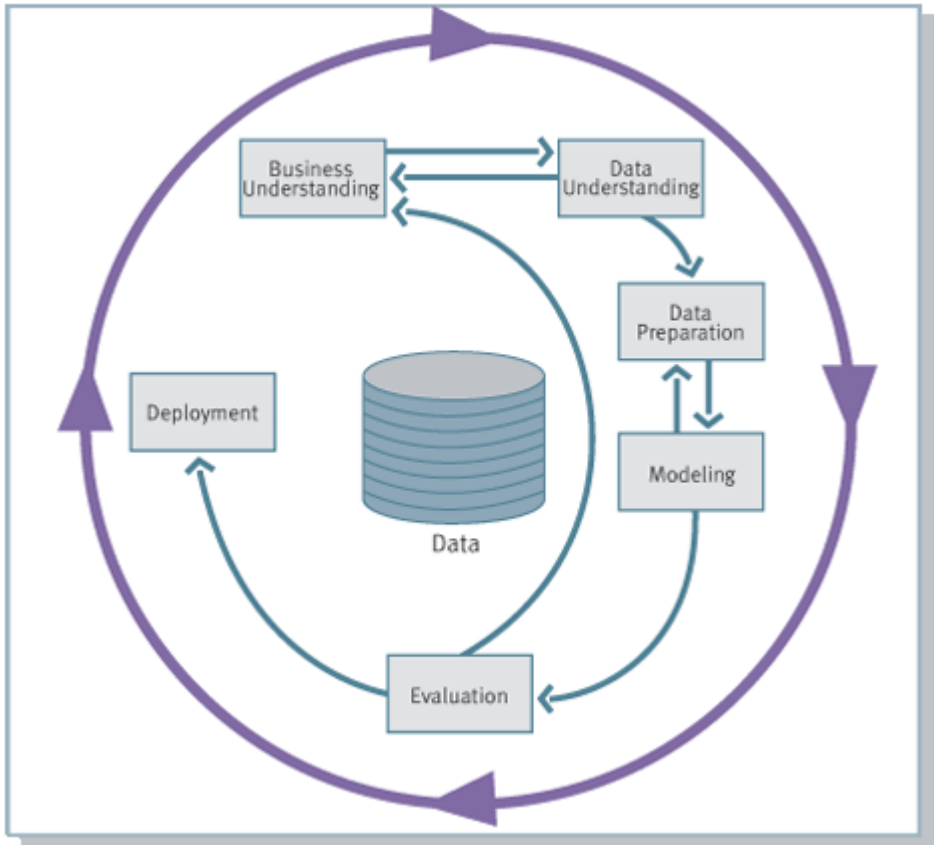
$$S = J\_measure * C \tag{12}$$

U izrazu (12) C predstavlja sumu jediničnih cijena svih proizvoda koji čine pravilo pridruživanja. Ovako definirani kriterij interesantnosti preko veličine C favorizira pravila koja sadrže pravila s većom jediničnom cijenom pravila, no u sebi implicitno preko *J-measure* sadrži i utjecaj ostalih relevantnih veličina koje karakteriziraju neko pravilo pridruživanja.

2.2 STANDARDI ZA DUBINSKU ANALIZU PODATAKA

Danas postoji više standarda koji definiraju proces dubinske analize podataka. Najpoznatiji i najviše primjenjivani standard je *CRoss Industry Standard Process for Data Mining* (CRISP-DM) [cri97] standard. Razlog njegove široke prihvaćenosti je njegova neovisnost o proizvođačima softvera za dubinsku analizu podataka. Pored CRISP-DM standarda neki od značajnijih standarda za dubinsku analizu podataka su Six Sigma [Pyz03] i SEMMA [sem05] standarda. SEMMA standard je industrijski standard od SAS Institute [sas05].

CRISP-DM je standard koji opisuje proces dubinske analize podataka. Osnovna značajka mu je da definira dubinsku analizu podataka kao iterativan proces koji se sastoji od više faza pri čemu se je moguće vraćati, ovisno o potrebama, u neku od prethodnih faza s ciljem postizanja postavljenog cilja. Slika 1 prikazuje CRISP-DM standard za dubinsku analizu podataka (slika je preuzeta s [cri97]).



Slika 1 – CRISP-DM standard

Proces dubinske analize podataka počinje s fazom razumijevanja problema kojeg želimo riješiti (*Business Understanding*). U ovoj fazi se daju odgovori na pitanja kao što su:

- Što želimo analizirati?
- Koji je kriterij uspješnosti analize?
- Koji su mogući dobici od analize?
- Koliko je okvirno vrijeme trajanja izgradnje modela za analizu?
- Koliki su troškovi pokretanja projekta za ovakvu analizu?

Ukoliko je poznata većina odgovora na prethodna pitanja može se krenuti dalje s analizom.

Faza koja slijedi je faza razumijevanja podataka o problemu (*Data Understanding*) koji nam stoje na raspolaganju. U toj fazi se prikupljaju svi relevantni podaci vezani uz problem koji se analizira, primjerice, za analizu potrošačkih košarica podaci koji nas mogu interesirati su: podaci s POS-ova, informacije o mjestu gdje su prikupljeni, vremenskom razdoblju u kojem su prikupljeni, demografski i ostali podaci o kupcima itd. U ovaj fazi se podaci opisuju i skupljaju na jedno mjesto kako bi poslužili u daljnjim koracima analize. Faza razumijevanja podataka ključna je i za odabir algoritma (odnosno tehnike modeliranja podataka) za dubinsku analizu podataka jer je odabir algoritma uvjetovan karakteristikama podataka. Vrlo često imamo situaciju kada nam za određeni tip analize na izboru stoji nekoliko tehnika modeliranja a rezultati primjene nekog od njih uvelike ovise o karakteristikama podataka koji ulaze u proces modeliranja. U ovoj fazi se vrši i ocjena kvalitete podataka tj. jesu li raspoloživi podaci dostatni za kvalitetnu analizu, jesu li prikupljeni podaci dovoljno kvalitetni da bi se iz njih mogli izvući korisne informacije itd. Za dubinsku analizu podataka vrijedi akronim GIGO (*Garbage In Garbage Out*) što znači da ako prikupljeni podaci ne odražavaju u dovoljnoj mjeri problem na koji hoćemo dobiti odgovarajuće odgovore, niti rezultati koje dobijemo iz tih podataka ne mogu dati korisnu informaciju o problemu. U tom slučaju treba nabaviti nove podatke ili podatke koji bolje odražavaju promatrani problem [pej04].

Faza pripreme podataka (*Data Preparation*) je bitna jer odabrana tehnika modeliranja podataka tj. algoritam koji se primjenjuje za otkrivanje skrivenih i složenih veza i uzoraka u podacima uglavnom diktira i format u kojem podaci moraju biti da bi analiza uopće i počela. Osim pretvaranja podataka u odgovarajući format, u ovoj fazi se vrši se i čišćenje podataka od eventualnih nekonzistentnosti, šuma, rješavanja problema *outliera* i slično. Faze razumijevanja i pripreme podataka za analizu su obično vremenski najzahtjevnije u cijelom procesu dubinske analize podataka i mogu zajedno uzimati i više od 80% ukupnog vremena trajanja cijele analize [pyl03][pyl99]. Pažljivo odrađivanje ovih dvaju koraka od ključne je važnosti za ispravnost konačnih rezultat analize.

Faza modeliranja (*Data Modeling*) se odnosi na samu primjenu odgovarajuće tehnike modeliranja odnosno algoritma na već prikupljene i pripremljene podatke. Vremensko trajanje ove faze obično ovisi o složenosti algoritma za analizu, ukupnoj količini

podataka koji se analiziraju, načina pristupa podacima koji se analiziraju, hardveru na kojem se vrši modeliranje itd.

Nakon što je modeliranje gotovo i imamo prve rezultate analize njihova se kvaliteta daje na ocjenu osobama koje poznaju domenu unutar koje se vrši analiza (*Evaluation*). Oni dobivene rezultate uspoređuju i ocjenjuju na temelju objektivnih i/ili subjektivnih kriterija.

CRISP-DM standard za dubinsku analizu podataka je iterativne prirode te dozvoljava povratak iz neke od faza analize u neku od prethodnih faza. Primjerice, ukoliko nakon evaluacije rezultata nismo zadovoljni dobivenim, možemo se vratiti u neku od prethodnih faza i pokušati učiniti neka poboljšanja u cijelom procesu i tako učiniti konačni rezultat boljim. Možemo se iz faze evaluacije rezultata/modela vratiti u fazu pripreme podataka pa pokušati odabrati prikladniju strategiju pripreme podataka, uklanjanja šumova, rješavanja problema nedostajućih vrijednosti (*missing values*) itd. Ako niti to ne daje rezultate, možemo se vratiti u fazu razumijevanja podataka i pokušati nabaviti više i boljih podataka za analizu.

Kad su ostvareni rezultati koji nakon evaluacije dobiju pozitivnu ocjenu, oni se primjenjuju za svrhu kojoj su namijenjeni (*Deployment*) – pokretanje ciljanih reklamnih kampanja, učinkovitije formiranja prodajnih kataloga, optimiranje cijena pojedinih grupa proizvoda, korištenje modela za procjenu kreditne sposobnosti itd.

3. APLIKACIJA ZA ANALIZU POTROŠAČKIH KOŠARICA

3.1 IZVEDBENE ZNAČAJKE

Aplikacija za analizu pravila pridruživanja je izvedena u programskom jeziku Java korištenjem Oracle-ove tehnologije za dubinsku analizu podataka.

Počevši od verzije 9i, Oracle RDBMS (*Relational Database Management System*) u sebi ima integriran poslužitelj za dubinsku analizu podataka (DMS – *Data Mining Server*). DMS pruža određene funkcionalnosti za dubinsku analizu podataka. Funkcionalnosti i izvedba DMS-a Oracle 10g [odm10g] i Oracle 9i [odm9i] baze podataka se u određenoj mjeri razlikuju. DMS Oracle 10g baze podataka pruža više mogućnosti za dubinsku analizu podatak u odnosu na DMS koji dolazi s Oracle 9i bazom podataka.

Najvažnija značajka DMS je da je integriran s bazom podataka pa se podaci i poslužitelj za dubinsku analizu podataka nalaze na istom mjestu. Zbog toga ne treba posebno programirati logiku za pristup podacima što je osobito važno u slučajevima gdje se analiziraju velike količine podataka. Takav je primjer upravo analiza transakcijskih podataka s POS-ova (POS – *Point of Sales*), gdje se dnevno ostvare stotine tisuća pa i milijuni transakcija. Kad se radi o tako velikim količinama podataka koje sve ne možemo odjednom „povući“ u radnu memoriju, potrebno je posebno voditi računa o pristupu podacima na disku (spora memorija), što može postati kritična točka izvođenja algoritma za analizu podataka. U slučaju integracije poslužitelja za dubinsku analizu podataka zajedno s bazom podataka manja je opasnost da će ovaj aspekt analize biti kritična točka u analizi te se možemo posvetiti drugim aspektima analize. Navedene činjenice su osnovni razlog zbog kojeg je DMS dobio prednost u odnosu na neke druge profesionalne programske pakete za dubinsku analizu podataka kao što su to SAS *Enterprise Miner* [sas05] ili SPSS [sps05] koji pružaju velike mogućnosti za dubinsko pretraživanje podataka, ali pretpostavljaju da su svi podaci koji se analiziraju istovremeno smješteni u radnoj memoriji što je u slučaju analize transakcijskih podataka s POS-ova neizvedivo.

DMS u sebi ima implementirane funkcionalnosti za različite tipove dubinske analize podataka [odm10g][odm9i]. Tim funkcionalnostima se pristupa preko Java API-ja. Kod Oracle 10g baze podataka osim Java API-ja postoji i PL/SQL API preko kojeg se isto tako može pristupati određenim funkcionalnostima DMS-a. Funkcionalnosti koje pružaju Java API i PL/SQL API nisu u potpunosti identične [odm10g].

U odnosu na CRISP-DM standard [cri97], funkcionalnosti za dubinsku analizu podataka koje pružaju navedena sučelja odnose se na faze pripreme i modeliranja podataka.

Aplikacija za analizu podataka s POS-ova (*Market Basket Analysis*) omogućava analizu velikog broja transakcija učinjenih na POS terminalima. Osnovne karakteristika takvih podataka su njihova velika količina, veliki broj različitih proizvoda koji se pojavljuju u

transakcijama te određene nekonzistentnosti u podacima koje trebaju biti obrađene na odgovarajući način kako bi se podaci uopće mogli analizirati ovim putem (primjerice razna storna, različiti načini prikazivanja iste transakcije itd.).

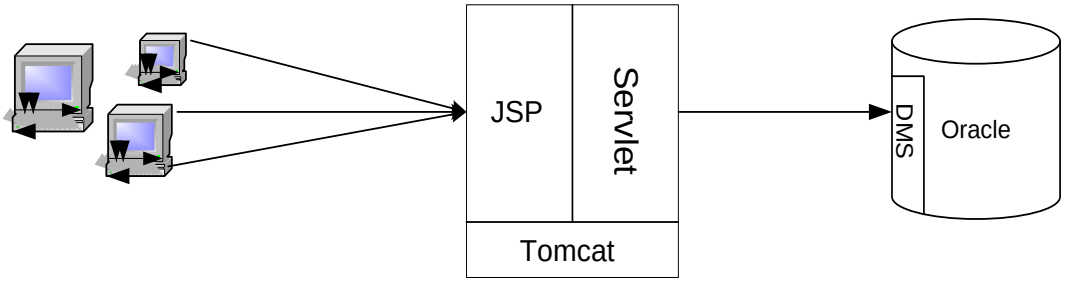
U testnom skupu podataka koji se odnosi na transakcije učinjene na samo jednom prodajnom mjestu tijekom 3 mjeseca imamo više od 300000 transakcija u kojima se pojavljuje više od 17000 različitih artikala. Otkrivanje pravila pridruživanja u podacima takvih karakteristika predstavlja poseban izazov s više aspekata:

- Tehničke karakteristike sustava trebaju biti takve da skrivaju kompleksnost analize i da korisniku omogući što jednostavniju upotrebu.
- Zbog velikog broja analiziranih transakcija i velikog broja različitih proizvoda koji se pojavljuju u transakcijama broj pravila koji se mogu pronaći može biti jako veliki. Stoga aplikacija treba imati mogućnost filtriranja rezultata analize tako da na kraju ostanu samo najinteresantnija pravila te mogućnost sortiranja tako dobivenih pravila po različitim kriterijima interesantnosti. U suprotnom postoji opasnost da se najinteresantnija pravila pridruživanja ne uoče između onih manje interesantnih.
- Korisnik treba imati i širi pogled na dobivene rezultate što podrazumijeva mogućnost povezivanja pronađenih pravila pridruživanja s drugim relevantnim informacijama kao što su mjesto gdje su podaci prikupljeni, vrijeme u kojem su prikupljeni i sl. Na taj način se uz informacije iz pravila pridruživanja dobivaju i dodatne dimenzije. Povezivanje dodatnih informacija s pravilima pridruživanja zahtijeva poseban informacijski model.

Proizvodi koji se pojavljuju u transakcijama slijede određenu taksonomiju (hijerarhijsko grupiranje proizvoda u potkategorije i kategorije). Aplikacija dozvoljava korisniku da vrši analizu pravila pridruživanja na različitim nivoima postojeće taksonomije. Svaki nivo analize daje određeni pogled na pravila pridruživanja.

Slika 2 prikazuje trorednu (*3-tier*) izvedbenu arhitekturu aplikacije koja se sastoji od:

- Prezentacijskog dijela
- Aplikacijskog dijela
- Podatkovnog dijela

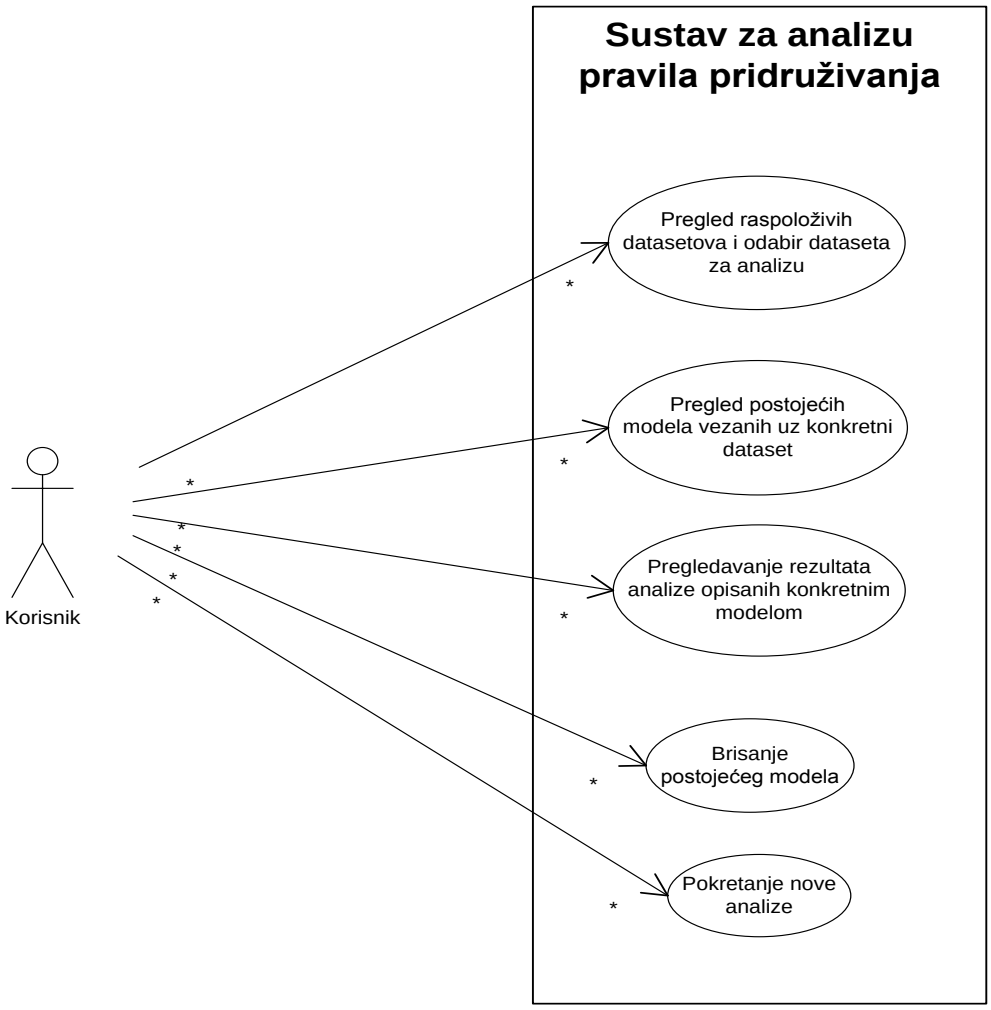


Slika 2: arhitektura aplikacije za analiziranje pravila pridruživanja

3.2 PREZENTACIJSKI RAZINA I UPORABA APLIKACIJE

Prezentacijska razina izvedena je korištenjem JSP (Java Server Pages) tehnologije. Zbog toga za korištenje aplikacije na korisničkoj strani je potrebno imati samo neki od Internet preglednika (Internet Explorer, Mozilla ...). JSP stranice su smještene u odgovarajući *web container* (u našoj implementaciji koristi se Tomcat Container [tom40]) koji je zadužen za generiranje dinamičkih web sadržaja.

Preko prezentacijskog dijela korisnik vrši interakciju sa sustavom za analizu pravila pridruživanja. **Slika 3** prikazuje UML dijagram slučajeva uporabe (*Use Case Diagram*) koji opisuje interakciju između korisnika i aplikacije (sustava):



Slika 3: Interakcija korisnika sa sustavom za analizu pravila pridruživanja

Funkcionalnosti koje sustav/aplikacija pruža korisniku su sljedeće:

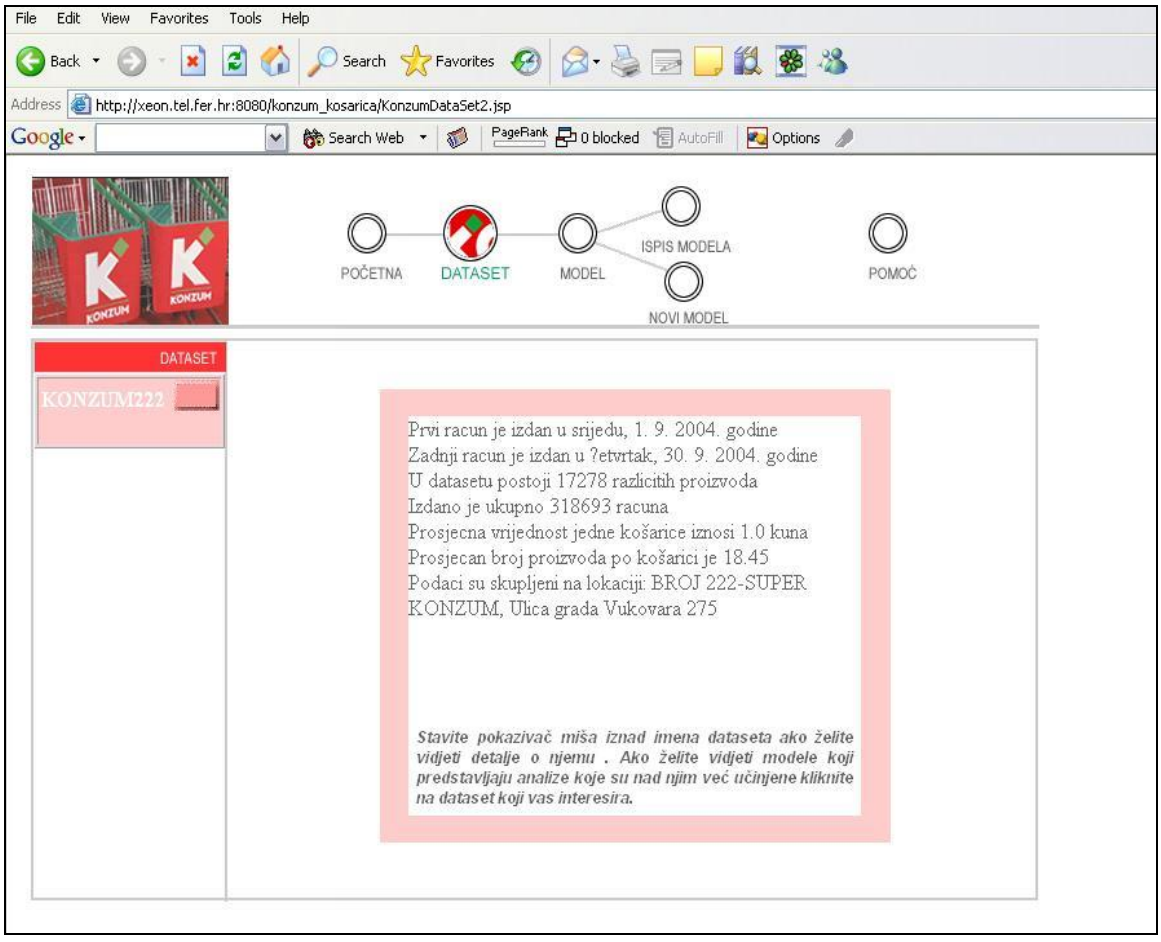
- Pregled raspoloživih skupova podataka i odabir skupa podataka za analizu (*dataset*)
- Pregled svih gotovih analiza pravila pridruživanja (modela) vezanih uz neki konkretni *dataset*
- Brisanje postojećih analiza odnosno modela koji reprezentira tu analizu
- Pregled pravila pridruživanje vezani uz neku konkretni model koji predstavlja analizu i prikaz pravila pridruživanja u obliku razumljivijem poslovnim korisnicima
- Kreiranje novog modela odnosno nove analize podataka s novim parametrima analize (značaj, pouzdanost, razina analize)

- On-line dokumentacija

Korisnik preko sučelja ima uvid u sve skupove podataka koji su raspoloživi za analizu. Uz svaki skup podataka dostupne su informacije koje opisuju *dataset* kao što su:

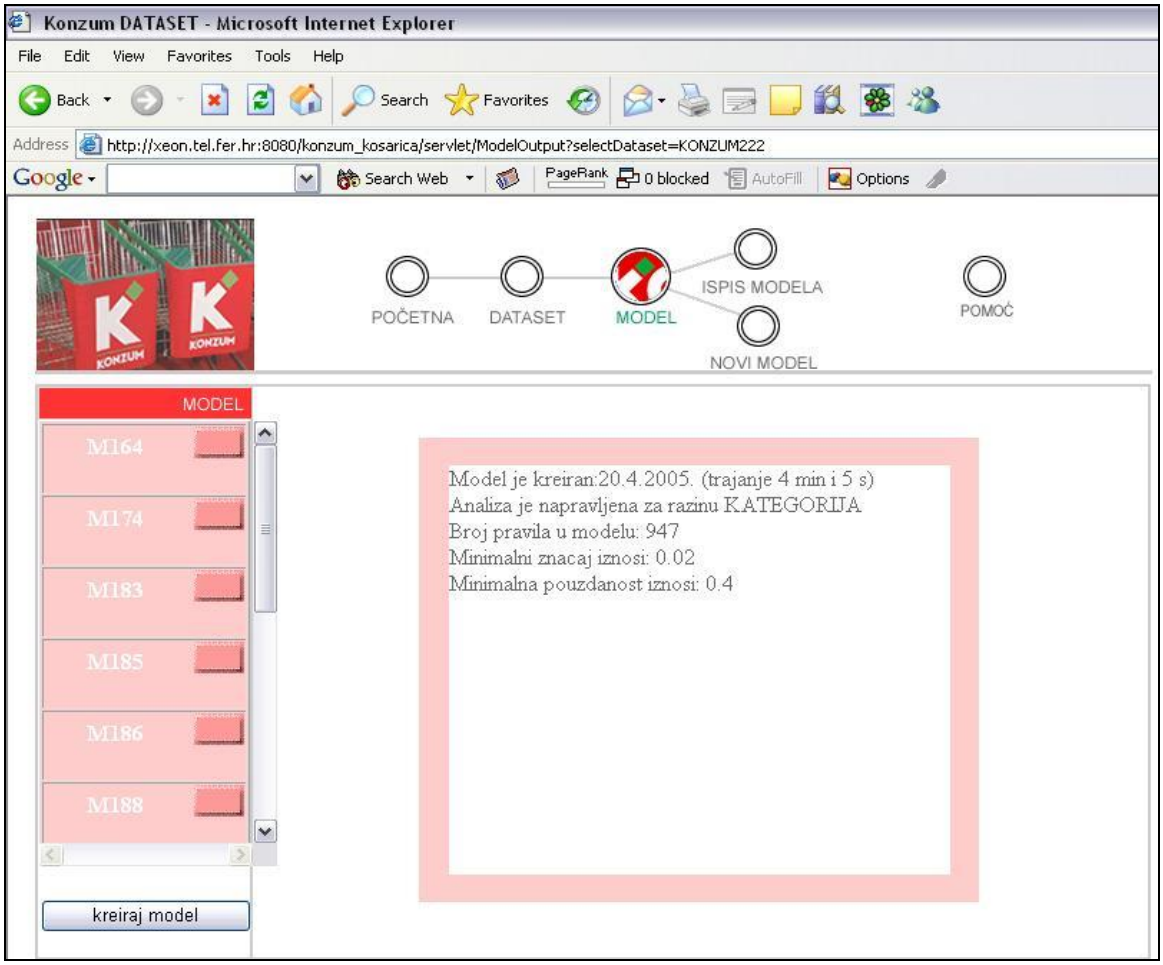
- Veličina *dataseta*: broj transakcija
- Broj različitih proizvoda koji se pojavljuju u transakcijama
- Prodajno mjesto na koje se *dataset* odnosi
- Vremenski period u kojem su podaci prikupljani
- Razni statistički podaci o *datasetu* (prosječna veličina potrošačke košarice, broj proizvoda u košarici itd.)

Slika 4 prikazuje informacije o *datasetu* (podacima koji se analiziraju) koje su dostupne korisniku aplikacije za analizu pravila pridruživanja.



Slika 4 – Podaci o datasetu

Za odabrani *dataset* korisnik ima pregled svih gotovih analiza koje se odnose na taj *dataset* (uključujući modele na svim razinama na kojima je moguće vršiti analizu). **Slika 5** prikazuje Ispis gotovih analiza (modela) za odabrani *dataset*.



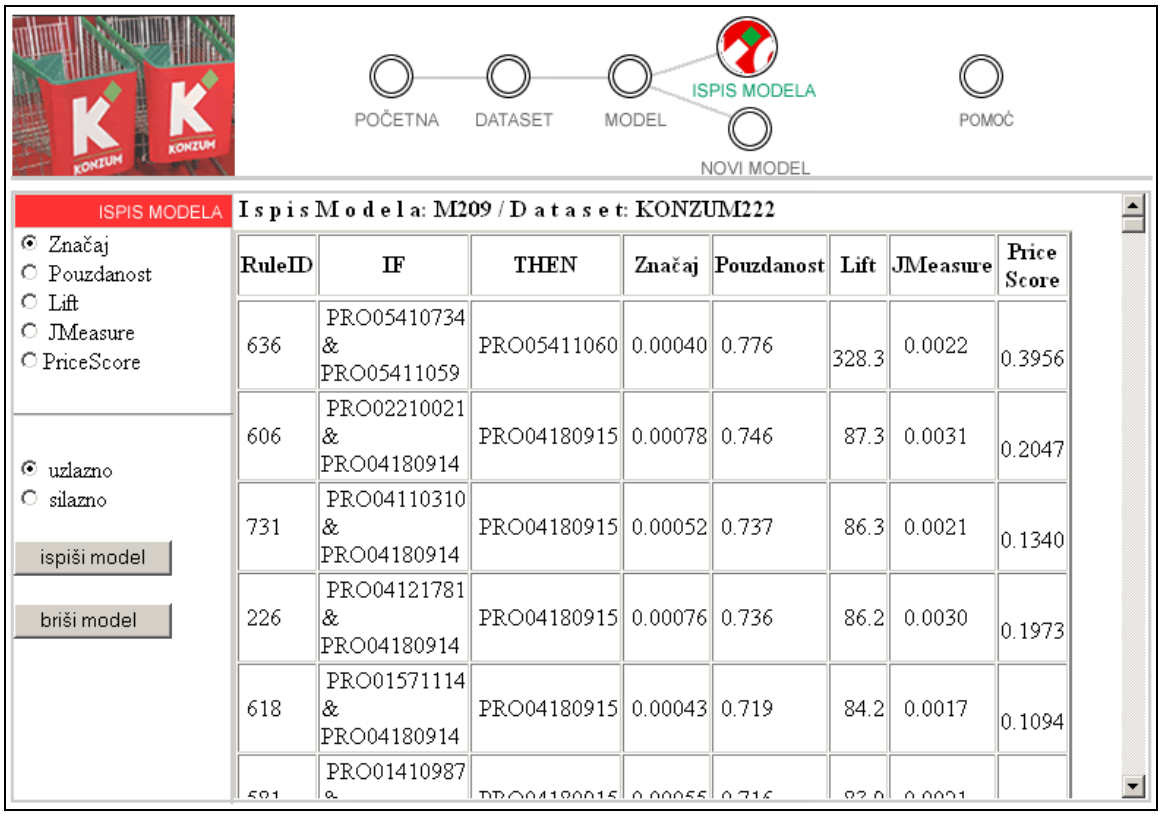
Slika 5 – Ispis gotovih analiza za odabrani dataset

Gotove analize koje su predstavljene odgovarajućim modelima nalaze se s lijeve strane (**Slika 5**). Svaki model ima jedinstvenu oznaku (npr. M164). Postavljanjem miša iznad pojedinog modela na lijevoj strani, na desnoj strani se ispisuju podaci koji su relevantni za pojedini model. Svaki model je određen s dva parametra: 'Minimalni značaj' i 'Minimalna pouzdanost'. Pored njih uz svaki model se nalazi informacija o datumu kad je kreiran odnosno kada je analiza učinjena, vremenu trajanja analize, nivou taksonomije na koji se analiza odnosi te broj pravila sadržanim u modelu. Ukoliko se želi kreirati novi model treba kliknuti na gumb 'kreiraj model'. Za ispis svih pravila pridruživanja sadržanih u nekom modelu treba kliknuti na taj model.

Za svaku gotovu analizu (model) može se dobiti ispis pronađenih pravila pridruživanja tako da se klikne na model koji nas interesira. Otkrivena asocijacijska pravila se mogu pretraživati po različitim kriterijima interesantnosti pravila pridruživanja. Za odabrani kriterij moguće je pravila sortirati (uzlazno i silazno) po različitim kriterijima:

- Po iznosu značaja (*support*)
- Po iznosu pouzdanosti (*confidence*)
- Po iznosu *lift*-a (poboljšanja)
- J-measure
- Po definiranom heurističkom kriteriju (*price score*) koji u obzir uzima i vrijednost proizvoda u košaricama

Na početku je model uvijek sortiran po pouzdanosti (*confidence*), kao na **Slika 6**. Veličina 'Price score' je definirana samo na najnižoj razini taksonomije (na razini pojedinačnih artikala).

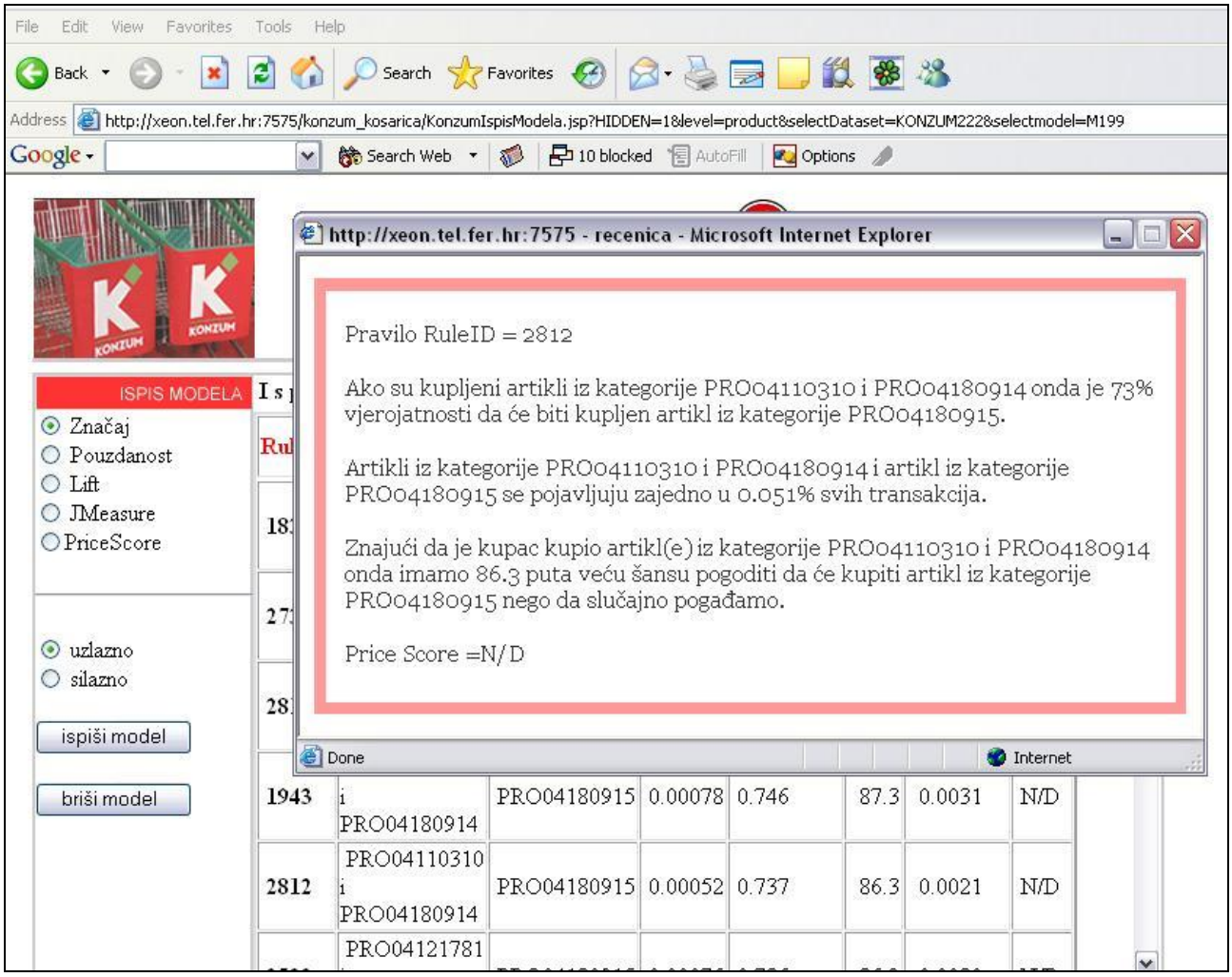


RuleID	IF	THEN	Značaj	Pouzdanost	Lift	JMeasure	Price Score
636	PRO05410734 & PRO05411059	PRO05411060	0.00040	0.776	328.3	0.0022	0.3956
606	PRO02210021 & PRO04180914	PRO04180915	0.00078	0.746	87.3	0.0031	0.2047
731	PRO04110310 & PRO04180914	PRO04180915	0.00052	0.737	86.3	0.0021	0.1340
226	PRO04121781 & PRO04180914	PRO04180915	0.00076	0.736	86.2	0.0030	0.1973
618	PRO01571114 & PRO04180914	PRO04180915	0.00043	0.719	84.2	0.0017	0.1094
581	PRO01410987 &						

Slika 6 – Ispis pravila pridruživanja za konkretni model

Jedno pravilo pridruživanja se sastoji od svog identifikatora (RuleID), lijeve strane pravila (IF), desne strane pravila (THEN), te veličina Značaj, Pouzdanost, Lift, J-measure i Price Score (samo na najnižoj razini hijerarhije) koje pobliže opisuju pravilo pridruživanja. Po veličinama Značaj, Pouzdanost, Lift, J-measure i 'Price Score' pravila pridruživanja se mogu sortirati. Ako se pravila pridruživanja žele sortirati po nekom od ponuđenih kriterija (gornji lijevi kut na **Slika 6**) dovoljno ga je odabrati i pritisnuti gumb 'ispiši model' i pravila pridruživanja će biti sortirana po odabranom kriteriju.

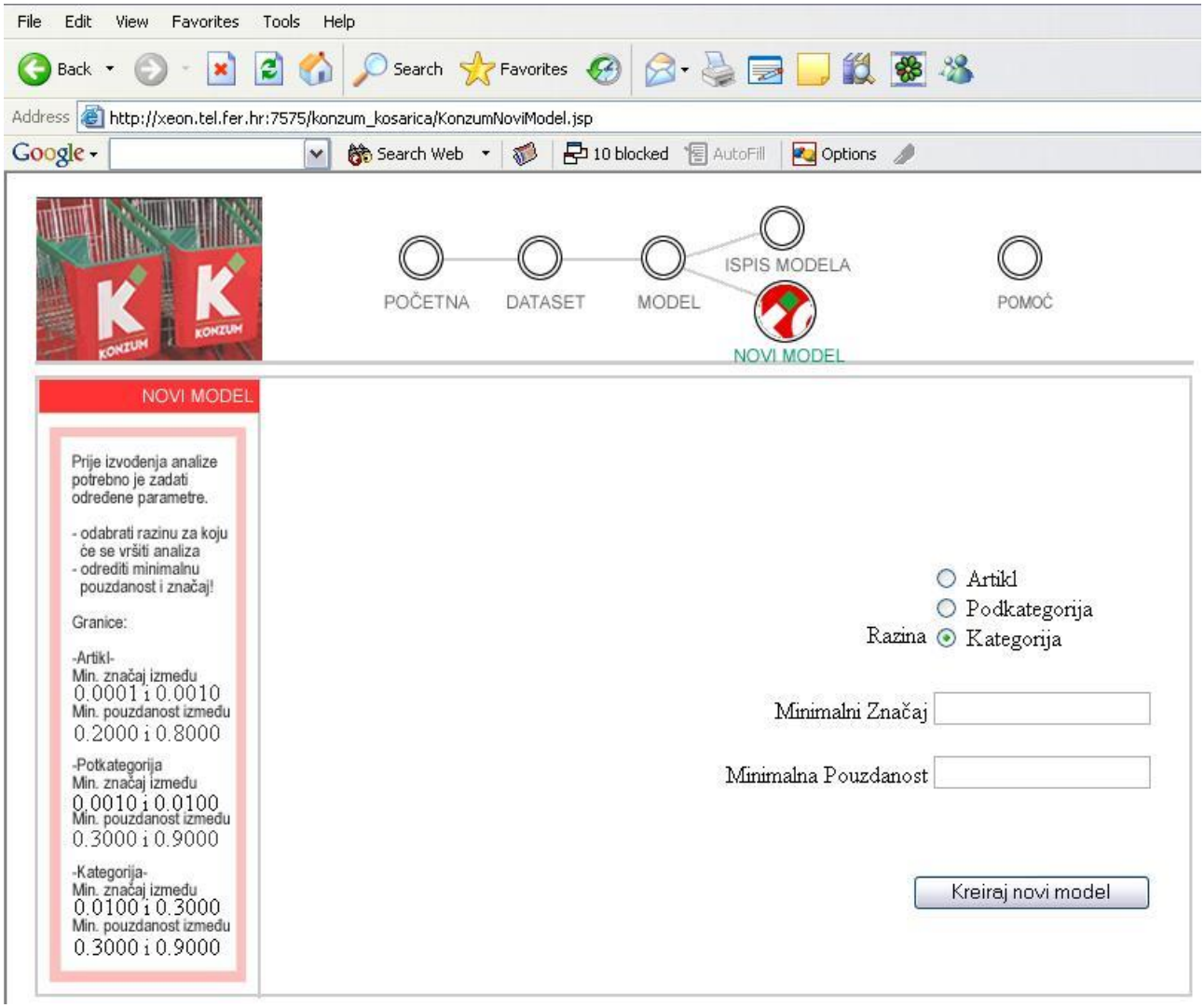
Osim u tabličnom obliku svako pravilo pridruživanja iz tablice se može prikazati u obliku kako se ono treba i čitati. Ako se klikne na ID pojedinog pravila dobiva se ispis pravila (**Slika 7**).



Slika 7 – Ispis pojedinačnog pravila pridruživanja

Korisnik ima mogućnost brisanja rezultata analize nakon što ih je pregledao.

Kod pokretanja nove analize korisnik mora unijeti odgovarajuće parametre za analizu. Radi se o minimalnim iznosima značaja i pouzdanosti te razini (prema odgovarajućoj taksonomiji) na kojoj se obavlja analiza.



Slika 8 - Pokretanje nove analize

Parametri koji mogu biti preneseni imaju svoje granice unutar kojih je moguće izabrati njihovu vrijednost. Minimalna pouzdanost pravila je u granicama [0,1]. Odabiranjem niže granice za pouzdanost konačni broj pravila raste pa odabiranje jako male minimalne pouzdanosti može rezultirati da u konačnici dobijemo jako veliki broj pravila od kojih većina ima malu pouzdanost. U tako velikom broju pravila lako se može dogoditi da se ne uoče neka moguće interesantna pravila pridruživanja.

Vrijednosti granica za minimalni značaj pravila ovise o tome na kojoj razini taksonomije se vrši analiza i određene su eksperimentalno na temelju testnih podataka. Ovisno o razini na kojoj se vrši analiza, postavljanje minimalnog značaja iznad te granice rezultira nepronalaskom niti jednog pravila pridruživanja koje ima pouzdanost veću od zadane. Obrnuto, postavljanje minimalnog značaja ispod određene granice bi rezultiralo generiranjem ogromnog broja pravila pridruživanja, a i drastično bi se povećalo vrijeme izgradnje modela koji predstavlja tu analizu. U ekstremnom slučaju, postavljanje jako male vrijednosti minimalnog značaja može rezultirati kolapsom cijelog sustava (jer

prostor mogućih pravila po kojem se vrši pretraživanje može postati nepretraživ u realnom vremenu). Zbog toga aplikacija ne dozvoljava unošenje parametara izvan određenih granica. **Slika 8** prikazuje na svojoj lijevoj strani granice unutar kojih je dozvoljeno postavljati vrijednosti minimalnog značaja i pouzdanosti za pojedini nivo analize. Granice za parametre se postavljaju i konfiguracijskoj datoteci (u kojoj se nalaze i ostale postavke za aplikaciju) te se po potrebi ovisno o karakteristikama podataka mogu mijenjati (što je potrebno činiti s posebnim oprezom i nakon potrebne analize).

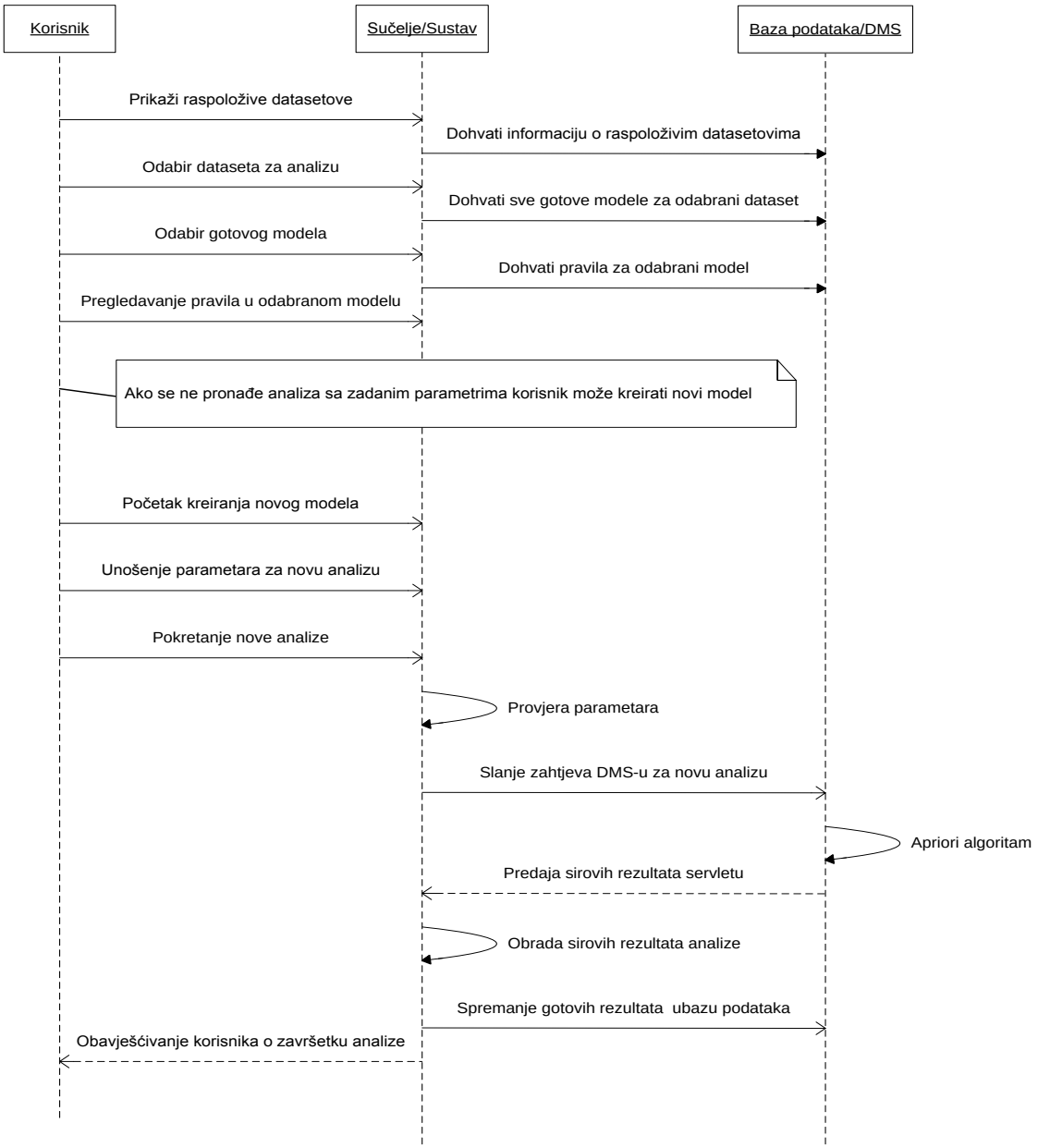
Ukoliko su uneseni parametri koji se poklapaju s vrijednostima neke od analiza koja je već učinjena nad tim *datasetom* (na istoj razini analize), aplikacija javlja korisniku da takva analiza već postoji i da je nije potrebno ponovo pokretati.

Ako je sve u redu, započinje analiza. Ovisno o hardveru računala na kojem se obavlja analiza, razini analize (na nižoj razini npr. razini pojedinačnih proizvoda, analiza traje dulje u odnosu na više razine), karakteristikama *dataseta* te unesenim parametrima analiza može trajati od nekoliko minuta pa do nekoliko desetaka minuta, pa čak i više. Analize koje su u tijeku su posebno označene i njihovi rezultati se ne mogu pregledavati dok ne završe.

Ako uneseni parametri ne odgovaraju definiranim granicama sustav korisniku daje odgovarajuću poruku i vraća ga na ponovno unošenje parametara analize.

3.2.1 Primjer analize

Slika 9 prikazuje mogući slijed aktivnosti prilikom pregledavanja postojećih modela (gotovih analiza) i pokretanja nove analize.



Slika 9 – primjer slijeda aktivnosti kod pregledavanje modela i nove analize

Slika 9 prikazuje samo najinteresantnije aktivnosti koje se događaju između pojedinih sudionika u sustavu.

Nakon pozdravne poruke korisnik aplikacije ima mogućnost pregledavanja svih dostupnih skupova podataka za analizu (*dataset*) (**Slika 4**). Uz svaki od *datasetova* su i osnovne informacije koje ih opisuju.

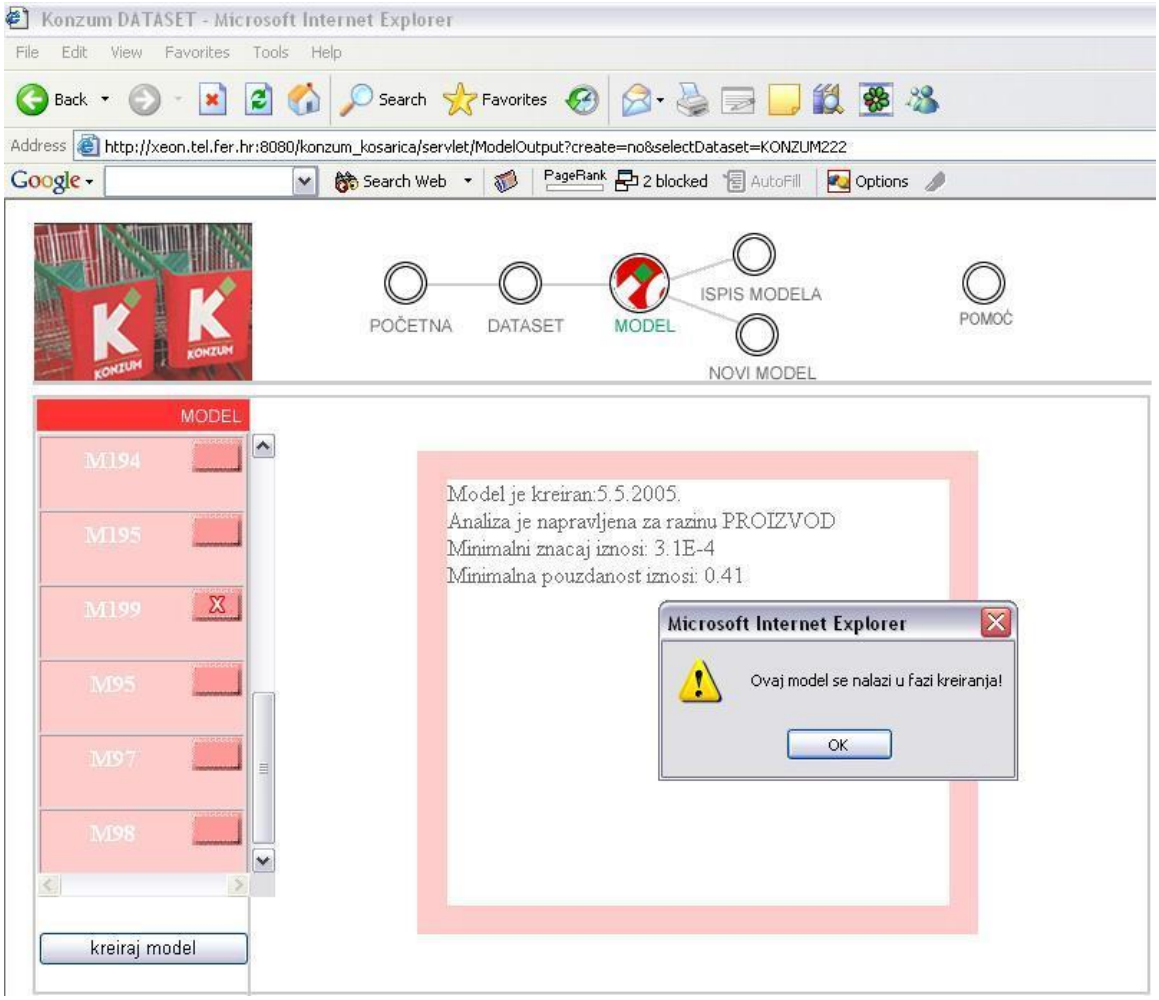
Nakon što se korisnik odluči koji će *dataset* analizirati, sustav dohvaća iz baze podataka informaciju o svim već učinjenim analizama. Korisnik može pregledavati osnovne informacije o gotovim analizama koje su predstavljene odgovarajućim modelima (**Slika 5**).

Kad se pronađe model s parametrima koji nas interesiraju, klikom na taj model dobije se prikaz svih pravila pridruživanja sadržanih u tom modelu (**Slika 6**). Ta pravila se onda mogu pretraživati (sortirati) po različitim kriterijima s ciljem pronalaska onih koji sadrže najinteresantnije informacije. Klikom na ID nekog pravila pridruživanja može se dobiti 'čitljiviji' ispis toga pravila koji prikazuje pravilo na način kako se ono treba čitati (**Slika 7**).

Ako prilikom pregledavanja gotovih modela nije pronađen model čiji nas parametri zadovoljavaju, korisnik može pokrenuti novu analizu. Prilikom pokretanja nove analize korisnik treba izabrati samo 3 parametra: 'Minimalni značaj' i 'Minimalnu pouzdanost' pravila te razinu na kojoj se analiza želi obaviti (**Slika 8**). Nakon toga sustav provjerava ispravnost unesenih parametra (granice i format) te postoji li već model koji ima iste parametre. Ako su granice u redu, a model s istim parametrima ne postoji, nastavlja se analiza. U suprotnom, korisnik dobiva odgovarajuću poruku.

Analiza se nastavlja tako da *servlet* šalje zahtjev za novom analizom zajedno s potrebnim parametrima DMS poslužitelju unutar Oracle baze podataka. U DMS poslužitelju se izvođenjem *apriori* algoritma dobivaju pravila pridruživanja. Pravila dobivena u DMS su sirovi rezultati analize te se šalju natrag *servletu* koji ih obrađuje. Rezultati iz DMS-a se potom filtriraju, računaju se potrebne veličine za svako pravilo pridruživanja itd. Nakon toga se ponovo spremaju u bazu podataka.

Dok traje analiza korisnik ne može pregledavati rezultate takvog modela, što je naznačeno odgovarajućom sličicom iznad oznake modela. Primjer daje **Slika 10**: vidi se oznaka da je model 199 u fazi kreiranja. Ako se u tom trenutku žele pregledati rezultati analize za taj model, korisnik dobiva odgovarajuću poruku.



Slika 10 – Analiza u tijeku

Kad nestane oznaka X iznad naziva modela, mogu se pregledavati rezultati nove analize. Trajanje analize ovisi o razini taksonomije na kojoj se analiza obavlja te vrijednostima minimalne pouzdanosti i značaja pravila. Analiza traje dulje što je niži nivo na kojem se vrši analiza i što su manje vrijednosti unesenih parametara.

3.3 APLIKACIJSKA RAZINA

Aplikacijska razina izvedena je korištenjem tehnologije Java Servlet [srv05]. Aplikacijska logika je smještena u istom *web container*-u kao i prezentacijska logika [tom50].

Aplikacijska logika obavlja višestruku zadaću:

- Skriva složenost analize od korisnika
- Komunicira s DMS poslužiteljem u Oracle bazi podataka, prenosi mu parametre potrebne za analizu, kontrolira tijek analize u DMS
- Preuzima sirove rezultate analize iz DMS

- Izračunava veličine za vrednovanje interesantnosti pravila
- Obrađuje rezultate dobivene iz DMS-a, vrši filtriranje pravila dobivenih u DMS-u
- Stvara konačne rezultate analize i sprema ih natrag u bazu podataka

Arhitektura sustava/aplikacije omogućava potrebnu skalabilnost odnosno paralelan rad više korisnika. Znajući da se radi o složenim i zahtjevnim analizama, na sustav su bili postavljeni posebni zahtjevi, koji su uspješno riješeni.

Obrada zahtijeva za analizu njegovo slanje prema DMS poslužitelju se odvija kroz niz složenih koraka koje aplikacijska logika skriva od korisnika. Prije izvođenja apriori algoritma u *servletu* se obavlja par koraka koji prethode slanju zahtijeva za izvođenje apriori algoritma na DMS-u.

Izvođenje apriori algoritma za pronalaženje pravila pridruživanja odvija se u DMS-u, koji je sastavni dio Oracle baze podataka te u određenom smislu možemo smatrati da je dio aplikacijske logike smješten i na strani baze podataka.

Komunikacija između *servleta* i DMS je sinkrona: cijelo vrijeme dok traje izvođenje apriori algoritma *servlet* prati u kojem je stadiju analiza na DMS. Kad je završeno izvođenje apriori algoritma u DMS-u, *servlet* to registrira i preuzima sirove rezultate analize iz DMS-a. Rezultati preuzeti iz DMS-a se u *servletu* obrađuju kako bi se pretvorili u konačni oblik pogodan za prezentiranje krajnjem korisniku. Obrada rezultata sastoji se od nekoliko koraka od kojih su najvažniji:

- Za svako dobiveno pravilo izračunavaju se veličine koje će pomoći pri određivanju njihove interesantnost a to su *lift*, *j_measure* i heuristička mjera interesantnosti pravila Price Score
- Filtriranje pravila
- Spajanje šifri proizvoda koji se pojavljuju u pravilima
- Pretvaranje pravila pridruživanja u oblik razumljiv korisnicima aplikacije

DMS server stvara pravila kojima je poznat samo iznos značaja i pouzdanosti. U situaciji kad analiziramo transakcije u kojima se pojavljuje mnogo različitih elemenata (proizvoda), broj pronađenih pravila pridruživanja čak i uz visoke iznose minimalnog značaja može biti jako velik. Pretraživanje takvog skupa pravila pridruživanja, oslanjajući se samo na iznose pouzdanosti i značaja, je jako otežan. Zbog toga se pravilima dodaju dodatni kriteriji koji olakšavaju pretraživanje pravila pridruživanja u potrazi za možebitnim interesantnim pravilima pridruživanja. U tom kontekstu pravilima se dodaju još tri veličine *lift* (poboljšanje), j*_measure*, te još jedan kriterij koji je domenski ovisan kriterij *Price Score* i koji uzima u obzir i vrijednosti proizvoda koji se pojavljuju u nekom pravilu.

Prema definiciji pravila pridruživanja X→Y i Y→X su dva različita pravila jer se općenito slijeva i desna strana pravila mogu imati različite značaje pa se stoga prema definiciji (tu treba staviti definiciju pravila pridruživanja) pouzdanosti tih dvaju pravila razlikuju (dok im je značaj isti jer se kod izračuna značaja pravila u obzir uzima i lijeva i desna strana pravila). Npr. neka su zabilježeni značaji 3 proizvoda (tj. njihove frekvencije pojavljivanja u transakcijama) sljedeće:

1. vino [0.1]
2. pivo [0.2]
3. riblja_pašteta [0.25]
4. vino & pivo [0.06]
5. riblja_pašteta & vino [0.05]
6. riblja_pašteta & vino & pivo [0.01]

U tom slučaju pravila a) i b) imaju sljedeće iznose značaja i pouzdanosti:

a) vino & pivo → riblja_pašteta [0.01, 0.17]

b) riblja_pašteta & vino → pivo [0.01, 0.2]

Dakle, formalno gledano, pravila koja uključuju iste proizvode ali s različitih strana pravila nisu ista pravila pridruživanja. No, općenito gledano, više smo zainteresirani za otkrivane interesantnih kombinacija proizvoda koje ne bismo očekivali na prvi pogled nego za statističke razlike između pravila koje uključuju iste elemente. Zbog toga aplikacija filtrira pravila tako da iz skupa pravila koji uključuju iste elemente filtriranjem ostavlja samo jedno pravilo i to ono koje ima najveći iznos veličine za *j-measure*. Tako se ukupni broj pravila koje korisnik dobije smanjuje između 30-40% u odnosu na broj pravila koji se dobivaju iz DMS-a čime se povećava vjerojatnost otkrivanja možebitnih interesantnih kombinacija proizvoda (ili grupa proizvoda).

Na kraju se sva pravila pretvaraju u oblik razumljiv krajnjem korisniku tj. šifre proizvoda se zamjenjuju s njihovim punim imenom i spremaju u bazu podataka. Korisnik na kraju ima pristup samo pravilima u bazi podataka (a ne čita ih iz DMS-a) te informacijama vezanim uz pravila koja su smještena u odgovarajući informacijski model.

Pored kreiranja novog modela u aplikacijskoj logici su implementirane funkcionalnosti koje omogućavaju korisnicima pregledavanje postojećih modela spremljenih u bazu podataka, brisanje postojećih, uvid u informacije vezanih u pojedine skupove podataka pripremljene za analiziranje te modele vezane uz te podatke.

3.4 PODATKOVNA RAZINA I PRIPREMA PODATAKA ZA ANALIZU

Kao podatkovnu razinu aplikacija (sloj za perzistenciju podataka) koristi Oracle 10g bazu podataka. Aplikacija koristi Oracle DBMS (*Database Management System*) *Enterprise Edition* i Oracle DMS (*Data Mining Server*) koji dolazi integriran s navedenom verzijom baze podataka i nije ga potrebno posebno instalirati.

Za korištenje DMS-a potrebno je kreirati posebnog korisnika Oracle baze podataka koji ima privilegije koristiti DMS. U shemu koja pripada tom korisniku dodajemo skupove podataka za analiziranje te još jedan dodatni skup podataka koji opisuje taksonomiju, odgovarajući šifrarnik te cijene pojedinačnih artikla. Ovaj dodatni skup podataka dijele svi skupovi podataka.

Za ispravno se funkcioniranje aplikacije mora kreirati još jedan korisnik Oracle baze podataka. U shemu koja odgovara ovom korisniku unose se metapodaci o podacima za analizu te o gotovim analizama. U ovoj shemi su smješteni i konačni rezultati analize.

Sve analize i ocjene kvalitete podataka su učinjene na temelju testnog skupa podataka koji je dobiven od Konzuma. Dobiveni skup podataka sadrži podatke prikupljene u razdoblju od 1.9.2004. do 28.11.2004 u jednom od Konzumovih dućana. Podaci obuhvaćaju preko 300000 transakcija s više od 17000 različitih artikla koji se u njima pojavljuju. Smatramo da je ovaj skup podataka dovoljno reprezentativan da se na njemu mogu dati dovoljno općeniti zaključci o podacima potrebnim za analizu pravila pridruživanja, a da su istodobno primjenjivi i na druge skupove podataka.

Podatke koji se nalaze u bazi čine podaci nad kojima se vrši analiza te metapodaci, "podaci o podacima". Metapodaci sustavno opisuju podatke o prodaji i omogućuju aplikacijskoj razini (razini poslovne logike) analizu nad tim podacima. Metapodaci obuhvaćaju podatke o *datasetovima* te o pojedinim modelima.

Podaci za analizu i metapodaci čine dvije zasebne logičke cjeline. Zbog veće sigurnosti oni su smješteni u dvije zasebne korisničke sheme iste baze podataka. Podaci o prodaji nad kojima se vrši analiza nalaze se u jednoj shemi, a metapodaci u drugoj.

3.4.1 Priprema podataka

Dodavanje novog skupa podataka za analizu tzv. *dataseta* vrši administrator sustava. Prije nego podaci budu dodani u sustav kako bi se analizirali oni moraju proći proces pripreme.

Testni skup podataka dobiven iz Konzuma sastoji se od dvije tablice. Prva opisuje transakcije učinjene na POS-ovima (*Point of Sale*) i to na razini svakog pojedinačnog kupljenog proizvoda (*line item*) a druga sadrži dodatne informacije o proizvodima koji se pojavljuju u transakcijama:

- taksonomija tj. hijerarhijsko organiziranje proizvoda u podgrupe i grupe
- puni nazivi proizvoda
- jedinicu mjere
- cijenu pojedinačnih proizvoda (po jedinici mjere)

Slika 11 prikazuje tablicu koja opisuje transakcije učinjene na POS-ovima.

DATUM	BROJ_RACUNA	SIFRA_ARTIKLA	KOLICINA
1.9.2004	310904069	01411805	0,52
1.9.2004	310904069	01400971	1,066
1.9.2004	310904069	04180450	1
1.9.2004	310904069	02053703	1
1.9.2004	310904069	02019096	0,304
1.9.2004	310904069	02053819	0,726
1.9.2004	310904069	02012371	1
1.9.2004	310904069	02012371	0,666
1.9.2004	310904069	04180450	1
1.9.2004	310904069	04170369	1
1.9.2004	310904069	02012371	-1
1.9.2004	310904069	03130019	1
1.9.2004	310904069	02330897	1
1.9.2004	310904069	04180450	1
1.9.2004	310904070	03185040	1
1.9.2004	310904070	03160267	1
1.9.2004	310904070	02231004	1
1.9.2004	310904070	02231004	1
1.9.2004	310904070	02334087	1
1.9.2004	310904070	02334087	1
1.9.2004	310904070	02334087	1
1.9.2004	310904070	02400098	0,527

Slika 11 – Podaci o transakcijama na POS-ovima

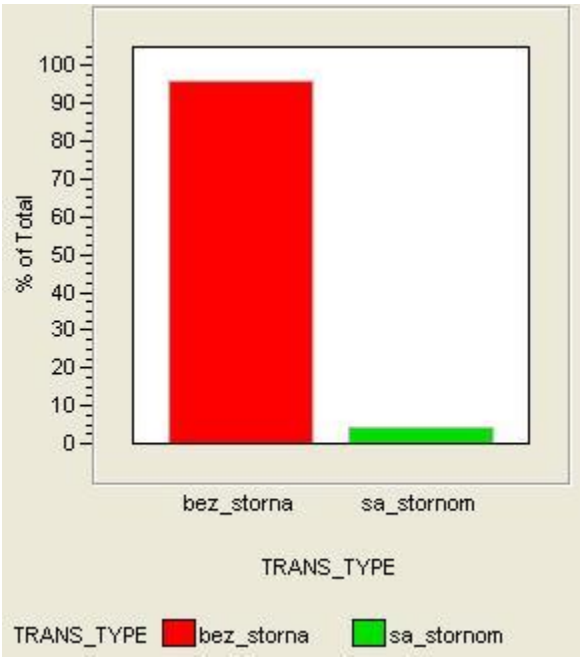
Prvi stupac iz tablice sa **Slika 11** označava datum kupnje, drugi broj računa odnosno TID (*Transaction ID*), treći stupac šifru kupljenog proizvoda i četvrti količinu kupljene robe (ostale informacije u ovom trenutku nisu bitne).

Slika 11 prikazuje uzorak iz baze podataka na kojem je vidljivo više problema koje treba riješiti prije nego podaci budu korišteni za analizu. Problemi koji su uočeni vezani su uz:

- Različite načine prikaza iste kupnje

- Storna tj. poništavanje kupnje
- Proizvode čija je jedinična količina izražena kategoričkom varijablom i proizvode čija je jedinična količina izražena numeričkom varijablom

Prvi problem koji treba riješiti je problem **storna** odnosno problem proizvoda čija je kupnja stornirana istodobno s transakcijom u kojoj ostvaruje kupnju ili kasnije (nakon što je kupio proizvod) iz bilo kojeg razloga. Ukupni udio transakcija u kojima postoje stornirani proizvodi je nešto manji oko 4.2% (**Slika 12**) što predstavlja nezanemarivi udio. Stoga problem storniranih proizvoda treba pozorno razmotriti.



Slika 12 – Udio transakcija sa i bez storna

Sa stajališta analize potrošačke košarice odnosno pravila pridruživanja treba razlikovati 2 tipa storna:

1. Storno kad kupac iz bilo kojeg razloga ne želi kupiti proizvod koji je zabilježen na POS-u (zbog krivog unosa šifre proizvoda na POS, zbog odustajanja kupca u trenutku kupnje itd.).
2. Storno kad kupac želi kupiti proizvod, ali je proizvod storniran iz nekog razloga (zbog krivo unesene količine, zbog uočene nepravilnosti itd.).

Proizvode koji se pojavljuju u prvoj skupini storniranih proizvoda treba eliminirati iz skupa podataka jer ne odražavaju stvarne potrošačke navike kupca. Storno tog tipa se uglavnom ostvaruju unutar transakcije za vrijeme kupnje i to tako da se unese isti proizvod ali s količinom koja ima negativnu vrijednost količine prvog unosa.

DATUM	BROJ_RACUNA	SIFRA_ARTIKLA	KOLICINA
1.9.2004	310904234	03070017	1
1.9.2004	310904234	02321022	3
1.9.2004	310904234	02321022	-3
1.9.2004	310904234	01410437	1,218
1.9.2004	310904234	02101952	1
1.9.2004	310904234	03171486	1
1.9.2004	310904234	02037346	0,412
1.9.2004	310904234	01480011	1
1.9.2004	310904234	04010162	4
1.9.2004	310904234	01480011	1

Slika 13 – Storno unutar iste transakcije

Slika 13 prikazuje transakciju označenu s brojem računa 310904234. Proizvod sa šifrom 02321022 je storniran. Potrebno je izbrisati proizvod sa šifrom 02321022 iz te transakcije (dakle i pozitivnu i negativnu vrijednosti količine).

U kratkim crtama, postupak uklanjanja ovakvih storna koji su se ostvarili unutar iste transakcije je taj da se vrši sumiranje po količini istog proizvoda i to po svakoj transakciji zasebno. Rezultat nakon sumiranja unutar pojedinačne transakcije po količini svakog proizvoda zasebno za transakciju 310904234 prikazuje **Slika 14**.

DATUM	BROJ_RACUNA	SIFRA_ARTIKLA	KOLICINA
1.9.2004	310904234	01410437	1,218
1.9.2004	310904234	01480011	2
1.9.2004	310904234	02037346	0,412
1.9.2004	310904234	02101952	1
1.9.2004	310904234	02321022	0
1.9.2004	310904234	03070017	1
1.9.2004	310904234	03171486	1
1.9.2004	310904234	04010162	4

Slika 14 – Nakon postupka sumiranja

Nakon toga treba obrisati unose u bazi podataka koji imaju vrijednost količine 0, čime se ujedno briše i stornirani proizvod (u konkretnom primjeru artikl sa šifrom 02321022). Time je riješen problem storna prvog tipa koji su ostvareni unutar iste transakcije, no ostaje problem onih storna (prvog tipa) koji su stornirani nakon što je proizvod kupljen. U tom slučaju prethodno opisani postupak uklanjanja storna neće eliminirati takva storna jer je storniranje izvršeno izvan transakcije u kojoj je zabilježena kupnja tog proizvoda. Za takva storna vrlo je teško (ako ne i nemoguće) napraviti heuristiku koja bi razlikovala takva storna od storna drugog tipa, a koja su isto ostvarena naknadno u zasebnim transakcijama. No za pretpostaviti je da je njihov udio statistički gledano zanemariv jer

općenito nije praksa vraćati proizvode kad su već kupljeni, osim ako se ne radi o nekom kvaru ili greški na robi, što podrazumijeva storna drugog tipa. Zbog toga se utjecaj storniranja tog tipa u naknadnim storniranjima može zanemariti.

Kod storna drugog tipa dovoljno je samo izbrisati informacije o storniranju, ali ne i proizvod jer je kupac na kraju kupio proizvod (ili ga je barem namjeravao kupiti). I tu razlikujemo dva slučaja. Jedan je kad je storno ostvaren u trenutku kupnje (primjer daje Slika 11). **Slika 11** u transakciji s brojem 310904069 prikazuje stornirani proizvod sa šifrom 02012371. Ako se pažljivo pogleda cijela transakcija, može se pretpostaviti da je storniranje nastupilo vjerojatno iz razloga što je blagajnik ili blagajnica prvi put krivo unio količinu proizvoda pa je stornirana u biti samo prva količina proizvoda. Naravno u ovom slučaju ne treba brisati cijeli proizvod već samo onaj dio koji se odnosi na storniranje. Postupak uklanjanja storna prvog tipa eliminirati će i ovakva storniranja, ali neće ukloniti i proizvod jer konačna suma količina neće biti 0 već neka pozitivna vrijednost. **Slika 15** prikazuje primjer transakcije s brojem 310904069 (zasjenjeni artikl je onaj artikl koji nas interesira) nakon procesa sumiranja po količinama.

DAT...	BROJ_RACUNA	SIFRA_ARTIKLA	KOLICINA
1.9.2004	310904069	04180450	3
1.9.2004	310904069	04170369	1
1.9.2004	310904069	04120656	2
1.9.2004	310904069	03220230	2
1.9.2004	310904069	03180804	1
1.9.2004	310904069	03130019	2
1.9.2004	310904069	02431606	0,478
1.9.2004	310904069	02330897	3
1.9.2004	310904069	02330448	1
1.9.2004	310904069	02233945	1
1.9.2004	310904069	02231025	1
1.9.2004	310904069	02102981	1
1.9.2004	310904069	02053819	0,726
1.9.2004	310904069	02053703	1
1.9.2004	310904069	02019096	0,304
1.9.2004	310904069	02012371	0,666
1.9.2004	310904069	01411805	0,52
1.9.2004	310904069	01408960	1,128
1.9.2004	310904069	01400971	1,066
1.9.2004	310904069	01400913	0,578
1.9.2004	310904069	01400100	0,056
1.9.2004	310904068	05400005	2

Slika 15 – Nakon postupka sumiranja po količini

Slika 15 daje do znanja da je količina uz stornirani proizvod 02012371 pozitivan broj 0.666 pa taj artikl u toj transakciji neće biti izbrisan kada budu brisani oni koji imaju 0.

Postupkom sumiranja po količini za svaki proizvod zasebno i za svaku transakciju zasebno znatno je smanjen broj transakcija sa zabilježenim storniranjem proizvoda. Udio transakcija sa storniranjem nakon postupka sumiranja sveden je na samo 0.6%.

Preostale transakcije sa stornom predstavljaju naknadna storniranja (oba tipa) učinjena nakon što je proizvod/artikl kupljen. Zbog njihovog malog broja u odnosu na ukupni broj transakcija, takve transakcije možemo isključiti iz daljnjeg postupka analize bez većih opasnosti narušavanja reprezentativnosti skupa podataka za analizu a s obzirom na analizu potrošačkih košarica. Naravno, analiziranje samih transakcija koje sadrže samo ovakva storna moglo bi otkriti zanimljive informacije.

Problem različitih načina prikazivanja iste transakcije je isto riješen postupkom sumiranja količina. Taj problem se odnosi na pojavu kada se npr. kupnja 6 piva može zabilježiti na više načina: jedan redak u bazi podataka s količinom 6, ili kao 6 posebnih redaka s količinom 1 itd.

3.4.2 Formatiranje podataka

DMS poslužitelj zahtijeva da podaci o transakcijama koji se analiziraju budu u točno određenom formatu. Taj format je opisan tablicom (**Slika 16**).

SEQUENCE_ID	ATTRIBUTE_NAME	VALUE
310904079	02311000	1
310904079	02320908	1
310904079	03220230	1
310904079	04010231	1
310904079	04141981	1
310904079	04149381	1
310904079	04163419	1
310904080	01310006	1
310904080	01310654	1
310904080	01310670	1
310904080	01400971	1
310904080	01401420	1
310904080	01401611	1
310904080	01401750	1
310904080	01402089	1
310904080	02037375	1
310904080	02050951	1
310904080	02053796	1
310904080	02055093	1
310904080	02100610	1
310904080	02140010	1
310904080	02180358	1

Slika 16 – Format podataka za DMS

Stupac SEQUENCE_ID odgovara stupcu BROJ_RACUNA (**Slika 11**), ATTRIBUTE_NAME odgovara stupcu SIFRA_ARTIKLA (**Slika 11**) i stupac VALUE odgovara stupac KOLICINA (**Slika 11**).

DMS uzima u obzir prilikom analize i različite količine proizvoda/artikla koje su kupovane u pojedinim transakcijama. To znači da DMS smatra pravila:

1. vino=1 & pivo=1 → riblja_pašteta = 1
2. vino=2 & pivo=4 → riblja_pašteta = 3

zasebnim pravilima pridruživanja (pravilo pod 1) čitamo: ako neko kupi 2 vina i 4 piva onda će kupiti i 3 riblje paštete).

Implementirani sustav za analizu pravila pridruživanja ipak ne koristi ovu mogućnost DMS-a. Razlog tomu je što je eksperimentalno utvrđeno da su prosječni značaji pravila ako se u obzir uzmu i količine kupljenih artikala ispod 0.0001 (0.01%). To je posljedica činjenice da se u transakcijama pojavljuje relativno veliki broj različitih artikala (u testnom skupu podataka preko 17000). Ukoliko bi se analize ograničavale na transakcije koje obuhvaćaju relativno mali broj artikala, tada bi eventualno imalo smisla razmatrati i ovu mogućnost što bi zahtijevalo i stanovite nadogradnje implementiranog sustava.

Zbog toga se vrijednost VALUE postavlja na 1 za svaki proizvod u transakcijama što znači da se promatraju samo pojavljivanje proizvoda, a ne i količine kupljenih artikala.

3.4.3 Dodatni podaci

Pored podataka o transakcija učinjenim na POS-ovima koji su opisani tablicom prikazanom na **Slika 11** sustav treba i dodatne podatke o svakom artiklu koji se pojavljuje u transakcijama koje se analiziraju. Te informacije se odnose na opis taksonomije, na pune nazive artikala, nazive grupa i podgrupa proizvoda prema definiranoj taksonomiji i jedinične cijene svakog od proizvoda. Navedene informacije se spremaju u posebnu tablicu (**Slika 17**).

SIFRA_ARTIKLA	SIFRA_GRUPE	SIFRA_PODGRUPE	IME_ARTIKLA	IME_GRUPE	IME_PODGRUPE	CIJENA
08144229	092	927	PRO08144229	CAT092	SUBCAT927	
08148068	092	927	PRO08148068	CAT092	SUBCAT927	
08152730	087	870	PRO08152730	CAT087	SUBCAT870	
08154587	087	872	PRO08154587	CAT087	SUBCAT872	
08303278	085	856	PRO08303278	CAT085	SUBCAT856	
06459927	092	926	PRO06459927	CAT092	SUBCAT926	
06513671	075	751	PRO06513671	CAT075	SUBCAT751	
06574367	076	767	PRO06574367	CAT076	SUBCAT767	
08015123	070	700	PRO08015123	CAT070	SUBCAT700	
08080042	071	718	PRO08080042	CAT071	SUBCAT718	
08131521	086	863	PRO08131521	CAT086	SUBCAT863	
05600005	087	872	PRO05600005	CAT087	SUBCAT872	
05775280	089	892	PRO05775280	CAT089	SUBCAT892	
06011487	084	841	PRO06011487	CAT084	SUBCAT841	
06080293	084	840	PRO06080293	CAT084	SUBCAT840	
06150593	084	843	PRO06150593	CAT084	SUBCAT843	
06190338	084	845	PRO06190338	CAT084	SUBCAT845	
02431460	015	152	PRO02431460	CAT015	SUBCAT152	
08401968	091	910	PRO08401968	CAT091	SUBCAT910	
08402370	091	912	PRO08402370	CAT091	SUBCAT912	
08402375	091	910	PRO08402375	CAT091	SUBCAT910	
08402377	091	912	PRO08402377	CAT091	SUBCAT912	
08402382	091	912	PRO08402382	CAT091	SUBCAT912	

Slika 17 – dodatne informacije o transakcijama

Svaki redak tablice (**Slika 17**) predstavlja jedan artikl. Imena stupaca određuju i njihovo značenje. Stupci SIFRA_ARTIKLA, SIFRA_GRUPE i SIFRA_PODGRUPE definiraju taksonomiju po kojoj su klasificirani artikli. Na osnovi tih informacija se omogućava korisniku da radi analize na različitim nivoima (pojedinačni artikl, podgrupa i grupa). Imena artikala, podgrupa i grupa omogućavaju ispis pravih naziva istih. Cijena omogućava definiranje heurističkog kriterija interesantnosti pravila.

4. ZAKLJUČAK

Transakcijski podaci s POS-ova predstavljaju vrijedan izvor informacija o potrošačkim navikama kupaca. No za izvlačenje svih korisnih informacija iz njih, nisu dovoljne samo OLAP analize (On-Line Analitical Processing) već je nužna primjena novih sofisticiranih tehnika analize kako bi se mogli dobiti odgovori na sva interesantna pitanja. Tehnike dubinske analize podataka (engl. *Data Mining*) su relativno nove metode analize podataka koje omogućavaju izvlačenje informacija koje na prvi pogled nisu očite niti su dostupne s klasičnim metodama analize podataka.

Analiza pravila pridruživanja je primjer takve analize. Implementirana aplikacija omogućava analizu velikih količina transakcijskih podataka. Aplikacija se temelji na provjerenim Oracle-ovim i Java tehnologijama čijom je primjenom dobiven robustan i skalabilan sustav za dubinsku analizu podataka. Robusnost se očituje u mogućnosti analize velikih količina podataka (reda veličine 1GB) a skalabilnost u mogućnosti istodobnog korištenja aplikacije od više korisnika te istodobne analize istih podataka na više nivoa odgovarajuće taksonomije. Aplikacija omogućava dobivanje informacija o tome koji se proizvodi (ili grupe proizvoda) zajedno kupuju unutar potrošačkih košarica zajedno sa svim ostalim informacijama koje omogućavaju vrednovanje otkrivenih pravila pridruživanja (značaj, pouzdanost, *lift*, statistička interesantnost pravila te domenski ovisan kriterij vrijednosti pravila pridruživanja). Informacije dobivene na ovaj način mogu biti značajne za razumijevanje potrošačkih navika kupaca što se onda može iskoristiti u CRM sustavima (*Customer Relatioship Managementu*), marketingu, prodaji, uređenju dućana itd.

5. LITERATURA

[odm10g] Oracle Data Mining – Concepts 10g Release 1 (10.1)

http://oraclelon1.oracle.com/docs/pdf/B10698_01.pdf

[odm9i] Oracle9i Data Mining – Concepts Release 9.2.0.2

http://download-uk.oracle.com/docs/pdf/A95961_02.pdf

[cri97] CRoss Industry Standard Proces for Data Mining official Web site

http://www.crisp-dm.org/

[agr94] Agrawal,R;Srikant,R:*"Fast algorithms for mining association rules in large databases",* Proceedings of the Twentieth International Conference on Very Large Databases (VLDB'94),pp. 487-499, 1994.

[han01] Hand,D.;Manilla,H.;Smyth,P.:*"Principles of Data Mining"*, MIT Press, 2001.

[hbd03] Group of Authors.: "*The Handbook of Data Mining*", Lawrece Erlbaum Associates, London, 2003.

[sas05] SAS Institute Inc. Official Web site

www.sas.com

[sps05] SPSS official Web site

www.spss.com

[pej04] Pejakovic,I.:"*Postupci dubinske pretrage podataka u sustavima poslovne inteligencije*", Magistarski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, 2004.

[pyl03] Pyle,D:"*Business Modeling and Data Mining*", Morgan Kaufmann Publishers, 2003.

[pyl99] Pyle,D:"*Data Preparation for Data Mining*", Morgan Kaufmann Publishers, 1999.

[odm04] Oracle Data Miner 10.1.0.1

http://www.oracle.com/technology/products/bi/odm/odminer_install.htm

[tom50] Apache Jackarta Project, Apache Tomcat

http://jakarta.apache.org/tomcat/

[srv05] J2EE, Java Servlet Technology

http://java.sun.com/products/servlet/

[sem05] SAS Enterprise Miner SEMMA

http://www.sas.com/technologies/analytics/datamining/miner/semma.html

[Pyz03] Pyzdek, T.: The Six Sigma Project Planner: *A Step-by-step Guide to Leading a Six Sigma Project Through DMAIC*. The McGraw-Hill Companies, 2003