UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Peter Konda

IZVEDBA PODATKOVNEGA RUDARJENJA V BANČNIŠTVU Z UPORABO METODOLOGIJE CRISP-DM

DIPLOMSKA NALOGA NA UNIVERZITETNEM ŠTUDIJU

Mentor: izr. prof. dr. Marko Bajec

Ljubljana, 2009

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01586/2009
Datum: 01.09.2009

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **SAŠO KNAP**

Naslov: **ZDRUŽITEV IN PRILAGODITEV METODOLOGIJ SCRUM IN EKSTREMNO PROGRAMIRANJE ZA ORGANIZACIJO RAZVOJA PROGRAMSKE OPREME V PODJETJU**

MERGING AND ADAPTATION OF SOFTWARE DEVELOPMENT METHODOLOGIES SCRUM AND EXTREME PROGRAMMING

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Za podporo razvoju programske opreme so danes na voljo številne metodologije. Posebej popularne so agilne metodologije, ki so navadno enostavnejše in lažje prilagodljive konkretnim potrebam. Ker je skoraj vsaka agilne metodologija v nečem dobra ali boljša od drugih, se je težko odločiti, kateri bi sledili. V okviru diplomske naloge preučite možnost združevanja agilnih metodologij ter njihovega prilagajanja konkretnemu podjetju. Kot primer vzemite metodologijo SCRUM in Extreme Programming.

Mentor:
prof. dr. Marko Bajec

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:
prof. dr. Franc Solina

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani/-a Peter Konda,

z vpisno številko 63010068,

sem avtor/-ica diplomskega dela z naslovom:

Izvedba podatkovnega rudarjenja v bančništvu z uporabo metodologije CRISP-DM

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom

  izr. prof. dr. Marka Bajca

- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela

- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne 10.12.2009 Podpis avtorja/-ice: _____

ZAHVALA

Zahvaljujem se svojemu mentorju izr. prof. dr. Marku Bajcu za pomoč pri izdelavi diplomske naloge. Sodelavcem iz NLB, d.d. se zahvaljujem za pomoč in podporo pri izvedbi projekta.

Staršema, bratu in sestri izrekam zahvalo za podporo in spodbudo na poti do diplome. Moji Andreji pa se zahvaljujem za spodbudo med izdelavo te diplome.

KAZALO

POVZETEK 1
ABSTRACT 2
1. Uvod 3
2. Mesto podatkovnega rudarjenja v računalniški znanosti 5
3. Razvoj podatkovnih skladišč 6
4. Metodologije za podatkovno rudarjenje 10
4.1. Uvod 10
4.2. CRISP-DM 11
4.2.1. Faze procesa 11
4.2.2. Analiza poslovanja 12
4.2.3. Analiza podatkov 13
4.2.4. Priprava podatkov 14
4.2.5. Modeliranje 15
4.2.6. Vrednotenje rezultatov 16
4.2.7. Uporaba rezultatov 17
5. Tehnike 19
5.1. Metode podatkovnega rudarjenja 19
5.1.1. Klasifikacija 19
5.1.2. Razvrščanje v skupine 20
5.1.3. Asociacije 20
5.1.4. Regresija in napovedovanje 21
6. Izvedba postopka podatkovnega rudarjenja 22
6.1. Opis problema 22
6.2. Izvedba procesa podatkovnega rudarjenja po metodologiji CRISP-DM 23
6.2.1. Analiza poslovanja 23
6.2.2. Analiza podatkov 24
6.2.3. Priprava podatkov 29
6.2.4. Modeliranje 32
6.2.5. Vrednotenje rezultatov 43
6.2.6. Uporaba rezultatov 44

7. Podatkovno rudarjenje z orodjem Weka 47
7.1. Uvod 47
7.2. Izvedba postopka 47
7.2.1. Analiza in priprava podatkov 47
7.2.2. Modeliranje 48
7.3. Primerjava rezultatov 49
7.4. Ugotovitve 50
8. Zaključek 52
9. Priloge 54
10. Literatura 55

SEZNAM KRATIC IN POJMOV

KRATICA	ANGLEŠKI POMEN	SLOVENSKI POMEN
	Cost benefit analysis	Analiza stroškov in koristi
CRISP-DM	CRoss-Industry Standard Process for Data Mining	Standardizirana metodologija za izvedbo procesa podatkovnega rudarjenja
DM	Data mining	Podatkovno rudarjenje
DMX	Data mining extensions	Strukturiran jezik za izdelavo modelov za podatkovno rudarjenje
DW	Data warehouse	Podatkovno skladišče
KDD	Knowledge discovery in databases	Proces pridobivanja znanja iz podatkovnih baz
KW	Knowledge worker	Znanjski delavec, ki na osnovi baze znanja in izkušenj ustvarja nove postopke, po katerih potem tudi deluje
OLAP	On-line analytical processing	Sistemi za hitro analiziranje podatkov
OLTP	On-line transactional processing	Sistemi za hitro procesiranje transakcij
PMML	Predictive model markup language	Izpeljava jezika XML za opisovanje strukture modelov
QA	Quality assurance	Oddelek za zagotavljanje kakovosti poslovnega procesa
RAD	Rapid application development	Metodologija razvoja programske opreme s hitrim prototipiranjem
RMSE	Root mean squared error	Kvadratni koren povprečne kvadratne napake
ROC	Receiver operating characteristic	Krivulja razmerja med občutljivostjo in senzitivnostjo za binarni klasifikator
ROI	Return on investment	Kazalnik donosnosti naložbe
SQL	Structured query language	Strukturiran jezik za izdelavo poizvedb v podatkovni bazi
SSIS	SQL Server integration services	Orodje za integracijo podatkov, ki je del platforme SQL Server 2008

POVZETEK

Podatkovno rudarjenje že več kot desetletje nastopa kot samostojna veja raziskovanja. Kljub temu se je prva standardizirana metodologija CRISP-DM, ki pokriva celoten proces podatkovnega rudarjenja, pojavila šele v začetku tega tisočletja. S formalizacijo procesa je postalo to področje zanimivo tudi za velike družbe, ki nove tehnologije praviloma sprejemajo z inercijo.

V diplomski nalogi sem predstavil podatkovno rudarjenje na aktualnem področju bančništva. Banka NLB, d. d., podobno kot večina velikih družb pri nas, že uporablja podatkovno skladišče in tehnologijo OLAP za analizo poslovanja. Te tehnologije pa ne omogočajo inteligentnega odkrivanja pravil ali vzorcev, zato podatkovno rudarjenje predstavlja logično nadgradnjo obstoječega sistema.

Diplomska naloga v uvodnih poglavjih opisuje umestitev podatkovnega rudarjenja v sodobno znanost in zgodovino razvoja podatkovnih skladišč. V nadaljevanju je podrobno opisana metodologija CRISP-DM in osnovne tehnike za reševanje problemov. Raziskovalni del naloge opisuje proces izvedbe podatkovnega rudarjenja, ki ga uporabljam za izračun naklonjenosti stranke k sklenitvi depozita. Proces, ki je potekal na podatkovni platformi SQL Server 2008, je izrazito iterativen s poudarkom na pridobivanju in analizi podatkov. Rezultate klasifikacijskih modelov sem med seboj primerjal grafično in z uporabo križnega preverjanja.

V orodju Weka sem na koncu ponovil postopek izdelave modelov in zapisal primerjavo zmogljivosti obeh orodij.

Ključne besede: poslovno podatkovno rudarjenje, metodologija CRISP-DM, SQL Server 2008, Analysis Services, Weka.

ABSTRACT

Data mining has been recognized as an independent field of research for more than a decade. Introduced in 2000 CRISP-DM is considered the first formal methodology that fully covers the process of data mining. Large companies now seek to incorporate this technology into their existing systems.

This thesis describes the uses of data mining in a bank. NLB, d. d. like most enterprizes in Slovenia established a data warehousing system. Using OLAP the employees can perform business analysis with ease, but may have problems finding complex patterns in the data. Therefore data mining represents a possible upgrade over existing systems.

The first few chapters introduce data mining and its place in modern science. Since data mining deals with data I included a brief history of data storage development. The next chapters contain a full description of CRISP-DM methodology and techniques for solving common business problems. The research part covers the data mining process in practice. The objective is to calculate a propensity score for each customer. This was done iteratively using the SQL Server 2008 database platform with strong emphasis on data loading and analysis. I compared the accuracy of different classification models using graphic representation and cross-validation.

I concluded the research by repeating the data mining process in Weka and writing a comparison of both tools.

Keywords: data mining in customer relationship management, CRISP-DM methodology, SQL Server 2008, Analysis Services, Weka.

1. Uvod

Pojem podatkovno rudarjenje (ang. data mining ali DM) se je pojavil v 90. letih prejšnjega stoletja. Temelji te tehnične stroke so bili postavljeni v 50. letih s pojavom strojnega učenja (ang. machine learning). Takrat so razvili prve algoritme za iskanje znanja, ki se v izboljšanih različicah uporabljajo še danes. Če smo zelo natančni, so prave temelje postavili že prvi statistiki z opredelitvijo osnovnih pojmov, kot so enota, populacija, vzorec in spremenljivka.
Definirati podatkovno rudarjenje ni preprosta naloga, saj praktično vsak avtor v strokovni literaturi uporablja svojo definicijo. V nadaljevanju jih navajam nekaj s strani slovenskih in tujih avtorjev:
»Podatkovno rudarjenje je interdisciplinarno področje, ki obsega statistiko, prepoznavanje vzorcev, strojno učenje in grafična orodja za podporo analizi podatkov in odkrivanje zakonitosti v njih.« [1]
»Podatkovno rudarjenje je analiza (velike) množice podatkov z namenom iskanja neznanih povezav in prikaz teh povezav na razumljiv in uporabniku koristen način.« [2]
»Podatkovno rudarjenje je proces samodejnega odkrivanja koristnih informacij v velikih podatkovnih zbirkah. Tehnike podatkovnega rudarjenja so sposobne preleteti velike baze podatkov z namenom iskanja novih in uporabnih vzorcev, ki bi sicer lahko ostali skriti.« [3]
»Podatkovno rudarjenje je proces analiziranja podatkov z namenom iskanja skritih vzorcev z uporabo avtomatiziranih metodologij.« [4]
»Iz operativnega vidika je podatkovno rudarjenje proces, povezan s podatkovno analizo, ki ga sestavlja zaporedje aktivnosti: definiranje ciljev analize, analiza podatkov, interpretacija in vrednotenje rezultatov.« [5]

Zgornje definicije povezujeta besedni zvezi prepoznavanje vzorcev in velika količina podatkov. Povezava ni naključna. Prepoznavanje vzorcev oziroma zakonitosti v majhni količini podatkov namreč ni nujno zanesljivo.

V poslovni domeni se podatkovno rudarjenje uporablja za obvladovanje tveganja, odkrivanje prevar, segmentacijo strank, pridržanje strank, navzkrižno prodajo, ipd. Gospodarske družbe z uporabo podatkovnega rudarjenja pričakujejo znižanje stroškov iz poslovanja in visoko donosnost naložbe (ang. return on investment).

Današnje transakcijske podatkovne baze omogočajo hranjenje podatkov velikosti več petabajtov (10^{15} B). Analizo podatkov omogočajo sistemi OLAP (ang. on-line analytical processing), vendar ti niso namenjeni samodejnemu napovedovanju in inteligentnemu iskanju vzorcev. Algoritmi za podatkovno rudarjenje so sposobni prebrskati veliko količino podatkov. Pri tem poiščejo uporabne vzorce, ki jih klasična analitična orodja niso zmožna. Poleg iskanja vzorcev so nekateri algoritmi sposobni napovedovati izide oziroma dogodke, kot so:

- Cena vrednostnega papirja v naslednjem mesecu
- Napoved temperature ozračja
- Število prodanih izdelkov v določenem starostnem segmentu (mladina, seniorji, ipd)

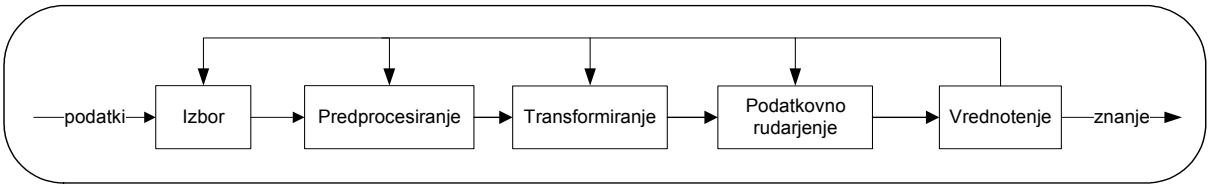
Pomembno področje je tudi iskanje pravil iz asociacij. Pogosto se omenja problem zlaganja izdelkov na police v trgovini tako, da bodo stranke v enem obisku kupile čim več stvari.

Moja diplomska naloga se usmerja na področje gospodarskih družb, bolj natančno bančništva. Cilj projekta v banki NLB, d. d. je za vsako stranko izdelati avtomatizirano napoved sklenitve depozita. Banka bo nato ponudbo posredovala samo tistim strankam, ki bodo depozit najverjetneje sklenile. S takšnim pristopom lahko znižamo stroške dela in materiala, ki bi bili pri naključnem izboru strank ali masovnem oglaševanju veliki.

Podatkovno rudarjenje raziskujem v poslovni domeni. Bolj kot podrobno analiziranje algoritmov je zato pomemben celotni delovni proces; od pridobivanja vhodnih podatkov do uporabe rezultatov modela, na primer v aplikaciji ali klicnem centru. Proces rudarjenja sem izvedel po metodologiji CRISP-DM, ki je tudi podrobno opisana. Pri tem sem uporabljal orodja in podatkovno bazo, ki so del platforme Microsoft SQL Server 2008. S programom Weka sem ponovil izdelavo modelov in na koncu opisal prednosti in slabosti uporabe obeh orodij.

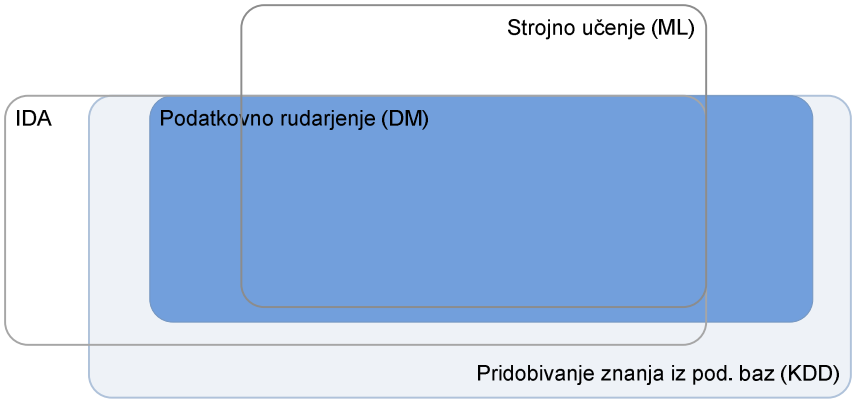
2. Mesto podatkovnega rudarjenja v računalniški znanosti

Podatkovno rudarjenje je del procesa, ki mu pravimo pridobivanje znanja iz podatkovnih baz (ang. knowledge discovery in databases ali KDD). Proces KDD se osredotoča na iskanje znanja v poljubnih vhodnih podatkih. Opira se na več strok, kot sta strojno učenje in statistika. Podatkovno rudarjenje je v bistvu eden izmed ključnih korakov procesa KDD. Pogosto se izraza DM in KDD uporabljata enakovredno. Pojavila sta se v 90. letih. Izraz podatkovno rudarjenje je postal bolj popularen v poslovnih krogih in novinarstvu.



Slika 1: Proces pridobivanja znanja iz podatkovnih baz [1]

Podatkovno rudarjenje je iterativni proces. To pomeni, da posamezne korake ponavljamo in s tem konvergiramo proti končni rešitvi problema. Večina metod podatkovnega rudarjenja izvira iz strojnega učenja.



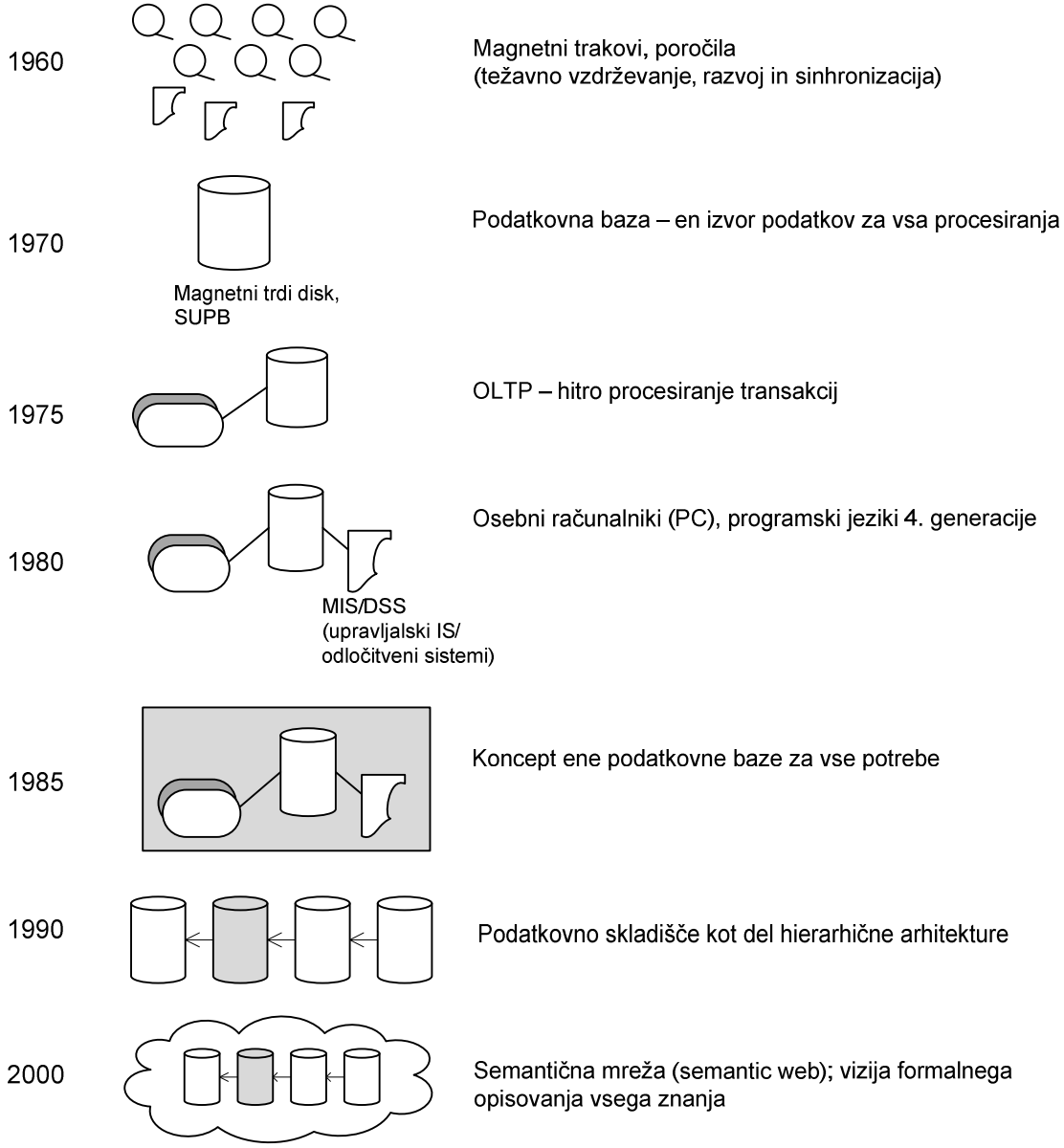
Slika 2: Vpetost podatkovnega rudarjenja med ostala področja [1]

Strojno učenje ni prava podmnožica podatkovnega rudarjenja, ker se opira na druga področja, ki jih podatkovno rudarjenje ne vsebuje, npr. teorija učenja in teorija izračunljivosti. Sorodno področje je tudi inteligentna analiza podatkov (ang. intelligent data analysis ali IDA). Tudi to področje se ukvarja z iterativnim procesom podatkovne analize. Od podatkovnega rudarjenja se razlikuje v tem, da uporablja metodologije in tehnike iz umetne inteligence in strojnega učenja.
V praksi velja, da imamo pri KDD/DM opravka z pridobivanjem informacij iz velikih podatkovnih zbirk, medtem ko je pri IDA količina podatkov manjša. OLAP denimo spada v analizo KDD/DM, ne pa tudi v inteligentno analizo podatkov.

3. Razvoj podatkovnih skladišč

Že ime nam pove, da je podatkovno rudarjenje nujno povezano z neko množico podatkov. Podatek kot neko dejstvo lahko zapišemo na mnogo načinov; od risanja po jamskih stenah do elektronskega shranjevanja na magnetnem disku. Za podatkovnega analitika, ki pripravi podatke za podatkovno rudarjenje, je pomembno, kako so ti podatki shranjeni in kako se do njih dostopa.

Industrija informacijske tehnologije se več kot pol stoletja ukvarja s prenosom podatkov iz fizičnih poročil, map in arhivov v elektronsko obliko. To jim zelo dobro uspeva in lahko rečemo, da se danes vsi poslovni procesi izvajajo elektronsko. Način elektronskega shranjevanja podatkov se je skozi leta bistveno spreminjal.



Slika 3: Evolucija elektronskega shranjevanja podatkov [6]

V 60. letih so prevladovali samostojni programi, ki so zapisovali podatke na magnetne trakove. Takšno shranjevanje podatkov je bilo ceneno, dostop do njih pa je bil, zaradi zaporednega branja, izredno počasen. Z naraščanjem števila magnetnih trakov so se pojavile težave s sinhronizacijo podatkov in kompleksnostjo programov.

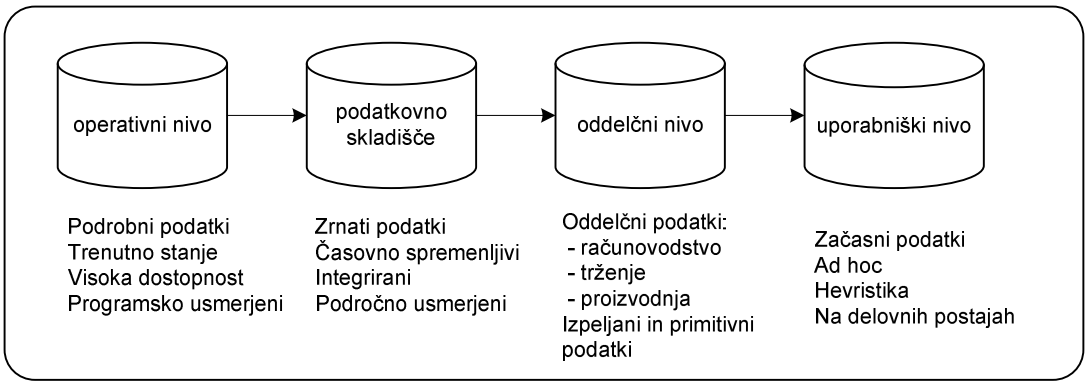
Razvoj in uporaba magnetnih diskov je rešila težave s hitrostjo zapisovanja in iskanja podatkov. V začetku 70. let so se začeli uporabljati sistemi za upravljanje podatkovnih baz (SUPB). Imeli so vgrajene funkcionalnosti za delo z zapisi, indeksiranjem in arhiviranjem, zato je razvoj programske opreme potekal bistveno hitreje. Sredi 70. let so se pojavili sistemi OLTP (ang. on-line transactional processing), ki so omogočali takojšen in varen dostop do podatkov. Znani primeri uporabe te tehnologije so: bančni avtomat, letalske rezervacije in upravljanje proizvodnje.

S pojavom osebnih računalnikov v 80. letih so uporabniki dobili novo vlogo – neposredno upravljanje proizvodnje in poslovanja. Upravljalski informacijski sistemi (ang. management information system) in kasneje odločitveni sistemi (ang. decision support systems) omogočajo izvajanje poslovnih odločitev, ki jih poslovodje sprejemajo pri vsakdanjem delu. Za poslovne odločitve potrebujemo informacije, ki so kompleksne in jih ne moremo pridobiti neposredno iz transakcijskih podatkov. Zato je potrebno izdelati tako imenovane vrtalne programe (extract programs), ki preko operacij ETL (ang. extract, transform, load) podatke naložijo in preoblikujejo v pravilno obliko. Težava je v tem, da količina teh programov s časom postane neobvladljiva. Temu pojavu pravimo pajkova mreža (ang. spider web). V velikih organizacijah je prihajalo do primerov, ko sta dva oddelka uporabljala različna programa za pridobivanje iste informacije. Ravnatelj je tako lahko dobil dve poročili o isti stvari s popolnoma drugačnimi številkami. S tem so odločitveni sistemi izgubljali verodostojnost. Da bi prišli do robustni sistemov, je bilo potrebno spremeniti arhitekturo. To spremembo predstavljajo podatkovna skladišča (ang. data warehouse ali DW) in njihova uporaba s strani sistemov OLAP.

Podatke lahko v osnovi razdelimo v dve skupini: primitivne in izpeljane. Primitivni podatki nastanejo z rednim delovanjem organizacije – operativni nivo. Izpeljani podatki so preračunani iz primitivnih in služijo v poslovnih odločitvah – analitični nivo. Navedimo nekaj razlik med obema tipoma podatkov:

- Primitivne podatke lahko posodabljamo. Izpeljane podatke lahko sproti preračunavamo, ne moremo pa jih direktno posodabljati.
- Primitivni podatki prikazujejo trenutno stanje. Izpeljani podatki običajno prikazujejo zgodovino.
- Primitivne podatke upravljamo s ponavljajočimi procedurah. Izpeljane podatke upravljamo hevristično z neponavljajočimi procedurami.
- Primitivni podatki podpirajo blagajniško delo. Izpeljani podatki podpirajo ravnateljsko delo.

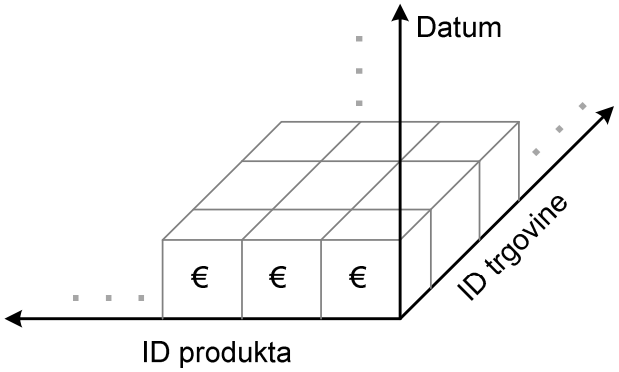
Bistvo nove arhitekture je fizična delitev obeh tipov podatkov. Prej smo imeli vse podatke v eni podatkovni bazi. Nov pristop pa zahteva za vsak nivo svojo podatkovno bazo.



Slika 4: Koncept arhitekture s podatkovnim skladiščem

Podatkovno rudarjenje se tipično izvaja na eni sami tabeli. V primeru izračuna verjetnosti sklenitve depozita, ki ga obravnavam v diplomski nalogi, to pomeni, da tabela vsebuje osnovne podatke o stranki in različna stanja: povprečno letno stanje na računu, povprečno stanje sredstev in obveznosti, povprečna plača oz. pokojnina, ipd. Vsa stanja predstavljajo agregirane vrednosti (povprečje, minimum, vsota) v določenem časovnem obdobju, npr. mesečno ali letno. Tabela, na kateri izvajamo podatkovno rudarjenje, predstavlja vrh strukture podatkov. Shranjena je v enem od mnogih podatkovnih skladišč, ki jih uporablja bančni sistem. Agregirana stanja, ki jih vsebuje omenjena tabela, prav tako dobimo iz podatkovnih skladišč, ta pa iz operativnih podatkovnih baz.

Namesto, da agregirane vrednosti računamo vsakič sproti z uporabo poizvedb, lahko uporabimo sisteme OLAP. Codd je leta 1993 zapisal 12 kriterijev, ki jim mora zadoščati podatkovna baza, da jo lahko uvrstimo v kategorijo OLAP [7]. Najbolj značilen kriterij je večdimenzionalni konceptualni pogled na podatke, kar si lahko predstavljamo kot kocko. Vsaka dimenzija kocke OLAP predstavlja atribut, ki nas pri analizi podatkov zanima. Kocka na sliki 5 prikazuje prodajo produktov po trgovinah v določenem obdobju.



Slika 5: Primer kocke OLAP s tremi dimenzijami

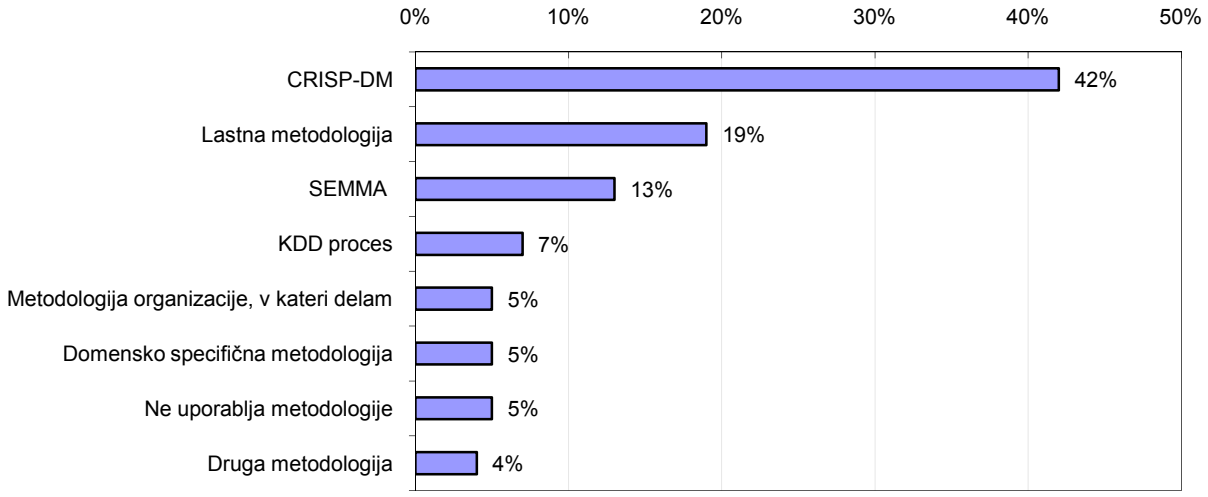
Z operacijo agregiranja (ang. roll up) lahko prikažemo prodajo v celotnem obdobju ali denimo po vseh izdelkih. Po drugi strani z operacijo vrtanja (ang. drill down) pridemo do konkretnega zapisa o prodaji.
Prednost uporabe sistemov OLAP v podatkovnem rudarjenju je v tem, da so podatki v kockah že pripravljeni za nadaljnjo analizo.

4. Metodologije za podatkovno rudarjenje

4.1. Uvod

Razvojna orodja za podporo podatkovnemu rudarjenju so prišle na trg dokaj pozno, v 90. letih prejšnjega stoletja. Govorimo o orodjih namenjenih znanjskim delavcem (ang. knowledge worker) [8]. Orodja, za uporabo katerih je bilo potrebno akademsko strokovno znanje, so obstajala že prej. Tako ne preseneča, da se je prva splošna metodologija za podatkovno rudarjenje pojavila šele konec prejšnjega stoletja.

V diplomski nalogi uporabljam metodologijo CRISP-DM, ki je v nadaljevanju podrobno opisana. Opisana metodologija še zdaleč ni edina. Praktično vsak proizvajalec orodij za podatkovno rudarjenje uporablja svoj procesni model. SAS Institute je vzporedno s programskim orodjem Enterprise Miner izdelal metodologijo SEMMA (Sample, Explore, Modify, Model, Assess). Sun je leta 2004 izdal JDM – objektni model in aplikacijski vmesnik (API) za standardizirano podatkovno rudarjenje. S tem so omogočili standardno povezovanje med različnimi orodji.
Anketa, ki jo je leta 2007 izvedla internetna skupnost KDNuggets, kaže pogostost uporabe metodologij za podatkovno rudarjenje [9].



Slika 6: primerjava metodologij za podatkovno rudarjenje (n=150)

Kljub temu, da 5% uporabnikov ne uporablja nobene metodologije, to ne drži, saj je po definiciji metodologija vse kar počnemo za dosego želenega rezultata [10]. Tudi podatkovnega rudarjenja se vedno lotimo po nekem postopku, formalno zapisanem ali z uporabo lastnega znanja.

Poudariti je potrebno, da se standardne metodologije, kot sta CRISP-DM in SEMMA, med seboj bistveno ne razlikujejo. Zaporedje faz je pri obeh enako.

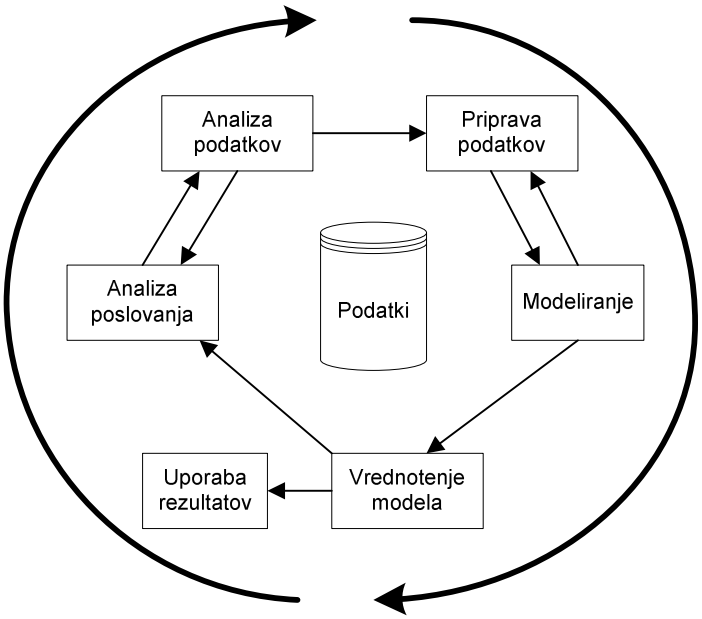
4.2. CRISP-DM

Temelje za nastanek CRISP-DM (ang. cross-industry standard process for data mining) standarda so postavila podjetja DaimlerChrysler, SPSS in NCR. Leta 1997 so oblikovala konzorcij s ciljem razviti standardni industrijski proces za podatkovno rudarjenje. Namenjen naj bi bil za uporabo v kateremkoli okolju, neodvisen od programskega orodja in gospodarskega področja.

Znanje o praktičnih delovnih procesih na tem področju ter mnenja o tem, kako le-te izboljšati, so pridobili na odprtih delavnicah [11]. Rezultat dela konzorcija je standard CRISP-DM 1.0, ki je nastal leta 2000 [12].

4.2.1. Faze procesa

CRISP-DM je splošno razumljiva metodologija za podatkovno rudarjenje. Razčlenjena je na šest razvojih stopenj. Stopnje so razdeljene na več opravil.



Slika 7: Razvojne stopnje metodologije CRISP-DM [1]

Notranje puščice prikazujejo povezanost med stopnjami. Zunanji krog simbolizira iterativno naravo samega podatkovnega rudarjenja.
Naslednja tabela prikazuje opravila posameznih stopenj in njihove izhodne dokumente.

Analiza poslovanja	Analiza podatkov	Priprava podatkov	Modeliranje	Vrednotenje rezultatov	Uporaba rezultatov
Določitev poslovnih ciljev • Poslovno ozadje • Poslovni cilji • Kriteriji uspešnosti **Ocena trenutnega stanja** • Prvine poslovnega procesa • Zahteve in domneve • Tveganja in slučaji • Stroški in koristi **Cilji podatkovnega rudarjenja** • Cilji • Kriteriji uspešnosti **Izdelava projektnega načrta** • Projektni načrt • Ocena potrebnih resursov	**Združevanje podatkov** • Poročilo o prenosu podatkov **Opis značilnosti podatkov** • Poročilo o značilnostih **Podrobna raziskava podatkov** • Poročilo o raziskavi **Ocena kakovosti podatkov** • Poročilo o kakovosti	**Izbor podatkov** • Racionalizacija zbirke **Čiščenje zapisov** • Poročilo **Izdelava podatkovne strukture** • Poročilo **Integracija zapisov** • Poročilo **Formatiranje vrednosti** • Poročilo	**Izbira tehnik** • Tehnika modeliranja • Predpostavke **Načrt modeliranja** • Pravila za izdelavo učne in testne množice **Izdelava modelov** • Nastavitev parametrov • Modeli • Opis modela **Ocenitev modelov** • Ugotovitve • Ponovni pregled parametrov	**Vrednotenje** • Ocena razmerja med uspehom rudarjenja in poslovnimi cilji • Potrditev modela **Pregled procesa** • Pregled poteka procesa **Določitev naslednjih korakov** • Seznam možnih akcij • Odločitve	**Načrt razvoja** • Načrt **Spremljanje in vzdrževanje** • Načrt **Končno poročilo** • Poročilo • Predstavitev projekta **Revizija projekta** • Dokumentacija

4.2.2. Analiza poslovanja

Pomemben del vsakega projekta poslovnega podatkovnega rudarjenja je razumevanje samega poslovnega procesa. Šele s pridobitvijo tega znanja in določitvijo poslovnih ciljev, je mogoče postaviti tudi cilje projekta. Dobro razumevanje poslovanja je tudi nujno potrebno za kasnejšo analizo podatkov.

4.2.2.1 Določitev poslovnih ciljev

Ta korak je ključen za celoten projekt, saj lahko napačno razumevanje ciljev pomeni, da bomo na koncu dobili sicer prave odgovore, a na napačna vprašanja. Poslovni cilj ima lahko določen vpliv, npr.:

- Zmanjšanje števila odhodov strank
- Povečanje prodaje nekega izdelka na določeni lokaciji

Lahko pa je zgolj informativen:

- Ali bi povišanje provizij na bankomatih povzročilo prestop strank k drugemu ponudniku?
- Za koliko lahko pričakujemo, da se bo povečal dnevni promet v trgovini ob podaljšanem odpiralnem času?

Dober poslovni analitik mora, poleg določitve ciljev, podati tudi pričakovano mero uspešnosti. Uspeh je lahko npr. zmanjšanje odhodov strank za 10 odstotkov. Pri tem mora biti cilj realno dosegljiv.

4.2.2.2 Ocena trenutnega stanja

V tem koraku analitik poda razpoložljivost prvin poslovnega procesa (resursov). Sem spadata predvsem programska oprema in zaposlenci, ki bodo potrebni za izvršitev nalog

podatkovnega rudarjenja. Zlasti pomembno je, da so na voljo vsi podatki v podatkovnih zbirkah. Izdelati je potrebno tudi pojmovnik, seznam projektnih tveganj in kako le-ta odpraviti ter analizo stroškov in koristi. (ang. cost-benefit analysis).

4.2.2.3 Določitev ciljev podatkovnega rudarjenja

Poslovni cilj je vedno mogoče prevesti v cilj podatkovnega rudarjenja. Primer: »Napoved števila izdelkov, ki jih bo kupila stranka na podlagi njenih preteklih nakupov, demografskih podatkov in cene izdelka«. V primeru, ko prevedba ni mogoča, je potrebno ponovno določiti poslovne cilje.

4.2.2.4 Izdelava projektnega načrta

Projektni načrt opisuje načrt za dosego ciljev podatkovnega rudarjenja. Vsebuje osnutke posameznih korakov, terminski plan, oceno tveganj in seznam potrebnih orodij in tehnik za izvedbo projekta.

4.2.3. Analiza podatkov

V tej fazi analitik ustvari zbirko podatkov, ki bo uporabljena za podatkovno rudarjenje. V tej zbirki mora poiskati določene zakonitosti ali hipoteze o informacijah, ki jih ta zbirka podatkov skriva.

4.2.3.1 Združevanje podatkov

Majhen nabor podatkov iz podatkovnih skladišč združimo v eni sami zbirki, ki je pravimo začetna. Analitik mora zabeležiti vse težave pri prenosu in integraciji v to zbirko. Primer težave je počasen dostop do podatkovne zbirke. Začetna zbirka mora odražati lastnosti vseh podatkov, nad katerimi kasneje izvedemo obdelavo.

4.2.3.2 Opis podatkov in njihove strukture

V tem koraku opišemo površinske lastnosti podatkov iz začetne zbirke. Med površinske lastnosti spadajo: število zapisov in polj v tabelah, opis polj in relacij med tabelami. Vprašanje, ki si ga v tem koraku zastavljamo je, ali so podatki začetne zbirke primerni za izdelavo modela. Primer: Starost osebe je pomemben atribut za modeliranje. Ker v začetni zbirki ni dovolj oseb s vneseno starostjo, je bolje izbrati drugačen vzorec zapisov ali pa atribut opustiti.

4.2.3.3 Podrobna raziskava podatkov

Pomemben del analize podatkov je odkrivanje atributov, ki bodo vključeni v podatkovno rudarjenje. Z osnovnimi tehnikami, kot so poizvedbe, vrtilne tabele in grafikoni, analitik odkrije določene zakonitosti in hipoteze. Te ugotovitve vplivajo na odločitve v naslednjih fazah projekta. Rezultat tega opravila je raziskovalno poročilo.

4.2.3.4 Ocena kakovosti podatkov

S tem opravilom podamo oceno kakovosti podatkov. Velike organizacije navadno zbirajo podatke v dolgem časovnem razponu. Informacijski sistemi se hitro spreminjajo, zato se

sčasoma zgodi, da stari zapisi nimajo vnesenih vseh atributov, ki jih potrebujemo pri podatkovnem rudarjenju. Teh zapisov pri pripravi podatkov ne upoštevamo. Težava ni nujno le v starih zapisih. Potrebno je odstraniti tudi zapise, ki močno odstopajo od pričakovanih vrednosti (npr. najstniki z zelo visokimi prihodki), saj ti lahko pokvarijo natančnost modela.

4.2.4. Priprava podatkov

V tretji fazi se izvajajo aktivnosti, ki pripeljejo do končne zbirke podatkov. Ta zbirka se uporabi v 4. razvojni stopnji za izdelavo modela.

4.2.4.1 Izbor podatkov

Na podlagi raziskav v prejšnji fazi se analitik odloči, kakšna bo končna zbirka podatkov uporabljenih za izdelavo modela. Izbor temelji na:

- Semantičnih kriterijih, npr. povezanost podatkov s cilji podatkovnega rudarjenje
- Tehničnih omejitvah, npr. podatkovni tipi in količina podatkov

Zgornji izbor je potrebno utemeljiti in razvrstiti atribute po pomembnosti za podatkovno rudarjenje.

4.2.4.2 Čiščenje podatkov

Da bodo rezultati podatkovnega rudarjenja čim boljši, morajo biti tudi podatki dovolj informativni. Zaradi velike količine podatkov se ponavadi izbere najboljša podmnožica zapisov. Izbor se izvede v skladu s poročilom o kakovosti podatkov, ki smo ga napisali v prejšnji razvojni stopnji.

4.2.4.3 Izdelava podatkovne strukture

Tipično se podatkovno rudarjenje izvaja na eni sami (normalizirani) tabeli, v kateri so vsi pomembni atributi. Osnovne atribute smo določili že v prejšnjih opravilih. Sedaj je potrebno izdelati strukturo za tabelo. Ta korak navadno vključuje naslednje opravke:

- Posamezne atribute združimo v nove, bolj informativne, atribute. Primer: dolžina in širina tvorita površino
- Zvezne atribute pretvorimo v diskretne. Primer: višino osebe tipa *double* pretvorimo v tip *boolean* (0 – nizka, 1 – visoka)

4.2.4.4 Integracija zapisov

V tem koraku iz podatkovne zbirke določene v 1. opravilu (Izbor podatkov) zapise integriramo v tabeli izdelani v prejšnjem opravilu. Ta tabela bo uporabljena za izdelavo modela v naslednji razvojni stopnji. Ker imamo opravka z eno tabelo, integracija običajno vključuje združitev (agregacijo) podatkov. Primeri agregacij so:

- Vsota vseh nakupov stranke v letošnjem letu
- Vrednost povprečnega nakupa stranke v prejšnjem letu
- Najvišja vrednost delnice v 3-letnem obdobju

Organizacija lahko uporablja podatkovno skladišče, v katerem so podatki že agregirani v OLAP kockah, kar bistveno olajša delo v tem koraku.

4.2.4.5 Formatiranje vrednosti

Potem, ko smo podatke zbrali v eni tabeli, je navadno potrebno vrednosti atributov nekoliko prilagoditi orodju za podatkovno rudarjenje. Prilagoditve so lahko povsem tehnične, npr. odstranjevanje nedovoljenih znakov ali zamenjava praznih vrednosti. Prilagoditve so potrebne tudi tedaj, ko spremenimo poslovni cilj.

4.2.5. Modeliranje

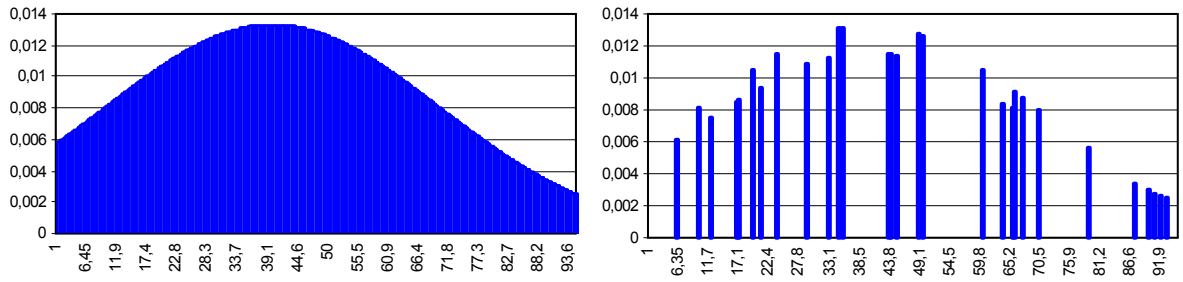
V tej fazi preskusimo več tehnik (algoritmov) za izdelavo modelov, ki so uporabne za obravnavani poslovni problem. Tehnike, kot so odločitvena drevesa, regresija, nevronske mreže, razvrščanje (ang. clustering), so podrobneje opisane v naslednjih poglavjih. Nekatere tehnike imajo specifične zahteve glede vhodnih podatkov, zato se je včasih potrebno vrniti v predhodno fazo.

4.2.5.1 Izbira tehnik

Tehnike, ki jih bomo uporabljali za izdelavo modelov v tem koraku evidentiramo. Popis vseh predpostavk prav tako opravimo v tem koraku. Predpostavke so odvisne od posamezne tehnike. Pri odločitvenih drevesih se na primer lahko odločimo, kako agresivno bo algoritem rezal odločitveno drevo.

4.2.5.2 Načrt modeliranja

Kvaliteto oz. točnost vsakega modela lahko preverimo z empiričnim testiranjem. Za testiranje se uporabljata dve množici, učna in testna. Prva se uporablja za učenje modela. Pod učenjem imamo v mislih izdelavo logičnih pravil, ki jih izdela algoritem za podatkovno rudarjenje. Natančnost delovanja pravil preverimo na testni množici.
Za učenje modela se uporablja izbrana množica podatkov. Ta lahko vsebuje vse ali le del zapisov – vzorec. Izbira vzorca je nujna takrat, kadar je, zaradi velike količine podatkov, učenje modela počasno in posledično drago. Podatki imajo praviloma neko verjetnostno porazdelitev. Enako porazdelitev morajo imeti tudi vzorčni podatki.



Slika 8: Primerjava velikosti vzorcev pri normalni porazdelitvi; levo celotna populacija, desno naključni vzorec z 2% populacije

Neka problemska domena ima lahko zelo majhen odstotek pozitivnih dogodkov. Takšen je primer pranja denarja, kjer število goljufivih transakcij navadno ne presega dveh odstotkov. Za učenje modela v teh primerih izberemo utežen vzorec. V primeru, da bi vzeli celotno množico transakcij, bi lahko algoritem izdelal model z napačnimi pravili.

4.2.5.3 Izdelava modelov

Kvalitetno parametrizirane algoritme v tem koraku uporabimo za izdelavo modela. Model je množica pravil, s katerimi rešujemo poslovni problem. Rešitev problema je lahko:

- Uvrstitev novih primerov (zapisov) v nek razred (klasifikacija)
- Napovedovanje gibanja vrednosti (regresija)
- Razvrščanje primerov v skupine (segmentacija)

V primeru napovedovanja nakupov v trgovini bi s pravili modela lahko dobili odgovor na vprašanje: Kakšno povišanje prodaje lahko pričakujemo v prihodnjem mesecu?

4.2.5.4 Ocenitev modelov

Model mora sedaj preveriti analitik s področja podatkovnega rudarjenja skupaj s poslovnimi analitiki. Poslovni analitiki dobro poznajo poslovanje in hitro opazijo neskladja v rezultatih. V splošnem mora model čim bolj ustrezati kriterijem uspešnosti, ki smo jih definirali v prvi fazi. Modele, izdelane z različnimi tehnikami, razvrstimo po uspešnosti in jih med seboj primerjamo.

Večina projektov podatkovnega rudarjenja predvideva večkratno uporabo iste tehnike za izdelavo modela. S spreminjanjem parametrov algoritma poskušamo z vsakim ciklom povečati njegovo natančnost.

4.2.6. Vrednotenje rezultatov

V predhodni fazi je bila pozornost usmerjena v natančnost in kakovost modela. Sedaj je potrebno oceniti, v kolikšni meri je model usklajen s poslovnimi cilji. Zanima nas, kako dobro model rešuje poslovni problem na novih podatkih. Delovanje modela preskusimo na čim večji množici podatkov.

Natančno je potrebno popisati pomanjkljivosti pravil modela. Pri tem lahko odkrijemo nove informacije in začrtamo smeri razvoja. Poleg tega dobimo realnejšo oceno doprinosa procesa podatkovnega rudarjenja.

Na koncu je izdelamo povzetek procesa in mnenje o tem, v kolikšni meri smo dosegli zastavljene poslovne cilje.

4.2.6.1 Pregled procesa

S temeljitim pregledom celotnega procesa lahko odkrijemo dejavnike, ki so ostali skriti. V tem koraku odgovorimo na vprašanja oddelka za zagotavljanje kakovosti (ang. quality assurance ali QA), kot so:

- Ali smo pravilno izdelali model?
- Ali bodo vhodni atributi modela uporabni v bodočem razvoju?

4.2.6.2 Določitev naslednjih korakov

V tem koraku se projektni vodja odloči, ali je rezultate možno implementirati, na primer v konkretni aplikaciji. V primeru, da se odloči negativno, je potrebno ponoviti določene faze. Ponavljanje faz je bistvo iterativne narave podatkovnega rudarjenja.

4.2.7. Uporaba rezultatov

Z izdelavo modela projekta še ni konec. Pridobljeno znanje mora biti organizirano in predstavljeno tako, da ga lahko razume in uporabi naročnik projekta. Znanje modela se lahko uporabi v odločitvenem procesu organizacije. Primera uporabe sta:

- Prikaz uporabniku prilagojenih spletnih strani v realnem času
- Izdelava prilagojenih trženjskih akcij

Rezultate (napoved, klasifikacija), pridobljene z uporabo pravil modela, lahko prikažemo v preprostem poročilu ali preglednici. Model lahko uporabimo tudi kompleksno, na primer z implementacijo v vseh poslovnih funkcijah: kadrovski, nabavni, proizvajalni, prodajni, itd.
Naročniku projekta ni potrebno poznati podrobnosti procesa, mora pa vedeti, kako pravilno uporabiti izdelane modele. Pri tem so ključni koraki: načrtovanje razvoja; spremljanje in vzdrževanje; končno poročilo; revizija projekta.

4.2.7.1 Načrtovanje razvoja

Ugotovitve iz faze vrednotenja uporabimo za izdelavo načrta o uporabi modela. Načrt mora vsebovati potrebne korake razvoja in njihovo implementacijo. Zanima nas tudi, kako bomo spremljali poslovne koristi podatkovnega rudarjenja in natančnost delovanja.

4.2.7.2 Strategija spremljanja in vzdrževanja

Ti dve aktivnosti sta pomembni, kadar se podatkovno rudarjenje uporablja dnevno v poslovanju organizacije. Posebno pozornost je potrebno posvetiti vzdrževanju. S tem se hitro odzovemo na nenadne napačne rezultate modela. Za nadzorovanje poteka razvoja je potrebno izdelati natančen načrt spremljanja. Vprašanja, na katera moramo odgovoriti, so:

- Kako bomo nadzorovali natančnost rezultatov modela?
- Kakšni so kriteriji spremljanja, na primer: veljavnost rezultatov, prag natančnosti modela, novi podatki, ipd

Pri tem moramo upoštevati specifike glede na tip razvoja.

4.2.7.3 Končno poročilo

Pri koncu projekta projektni vodja skupaj z ekipo napiše končno poročilo. Glede na načrt razvoja, je lahko končno poročilo povzetek projekta in izkušenj ali pa obsežna predstavitev projekta in končnih rezultatov. Poleg končnega poročila stranki predamo še predstavitev, kjer so povzetki zapisani na strukturiran in razumljiv način.

4.2.7.4 Revizija projekta

Po končani implementaciji ocenimo, kaj je šlo narobe oziroma, kaj bi lahko naredili bolje. Napisati moramo revizijsko poročilo. Zapišemo izkušnje, ki so nastale med izvajanjem projekta; na primer zavajajoče pristope, pasti in namige o izbiri tehnik rudarjenja. Aktivnosti revizije so:

- Intervjuvanje pomembnih članov projekta o njihovih izkušnjah

- Raziskava o tem ali končni uporabniki uporabljajo rezultate podatkovnega rudarjenja. Ali so zadovoljni z rezultati? Kaj bi lahko delovalo bolje? Ali potrebujejo dodatno podporo?

V idealnem projektu naj bi revizijsko poročilo vsebovalo povzetke vseh poročil, ki so jih napisali vsi člani projekta.

5. Tehnike

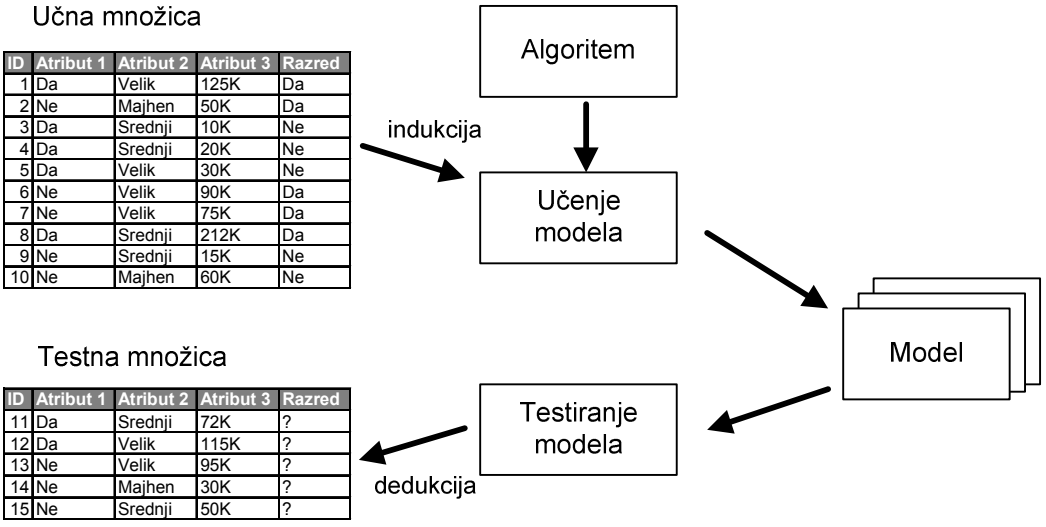
Algoritmi za podatkovno rudarjenje se delijo v dve skupini: nadzorovani in nenadzorovani. Pri nadzorovanih se odvisna spremenljivka izračuna na podlagi neodvisnih. Rečemo tudi, da ti algoritmi potrebujejo učitelja (odvisna spremenljivka), da se lahko učijo. Nenadzorovani algoritmi obravnavajo vse spremenljivke neodvisno. Takšni algoritmi se ne učijo na podlagi ciljne spremenljivke, temveč skozi iteracije konvergirajo proti cilju. V primeru segmentacije, je cilj stabilna ločnica med posameznimi skupinami.

V poslovni domeni se algoritmi za podatkovno rudarjenje uporabljajo za nabor poslovnih problemov. V nadaljevanju so najprej na kratko opisane metode podatkovnega rudarjenja in problemi, ki jih z njimi rešujemo.

5.1. Metode podatkovnega rudarjenja

5.1.1. Klasifikacija

Podatkovno rudarjenje se pogosto uporabljamo za klasifikacijo, to je uvrščanje objektov v razrede. Predpostavimo, da imamo objekt, ki ga opisujejo atributi (lastnosti). Atributi so diskretne ali zvezne neodvisne spremenljivke. Razred je odvisna diskretna spremenljivka. Vrednost razreda se določi iz vrednosti atributov. Naloga algoritma je uvrstitev objekta (primera) v razred.



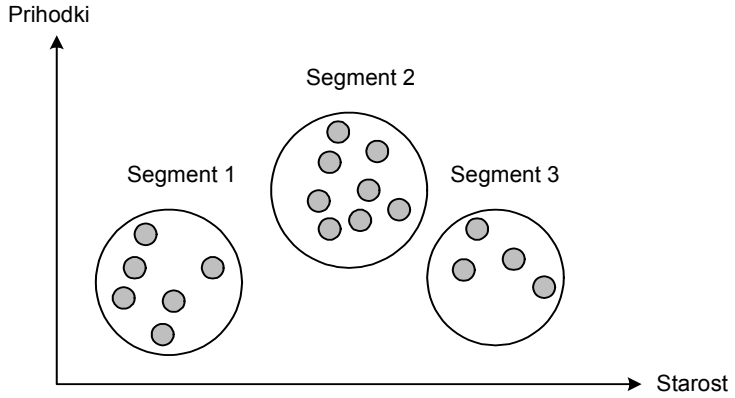
Slika 9: Postopek reševanja klasifikacijskega problema

Izbran algoritem iz vhodnih podatkov inducira pravila, ki so shranjena v modelu. Pravila nato testiramo na testnih podatkih, kjer preverimo kakovost pravil.
Poslovni problemi, kot so analiza prebega strank, obvladovanje tveganja in ciljno usmerjeno trženje, uporabljajo klasifikacijo. V primeru napovedi sklenitve depozita, vsako stranko uvrstimo v razred *cilj* (ang. target), ki ima dve možni vrednosti: da in ne. Vrednost razreda nam pove, ali bo stranka sklenila depozit ali ne.

Algoritmi, ki jih uporabljamo za klasifikacijo so: odločitvena drevesa, nevronske mreže, regresija in naivni Bayes.

5.1.2. Razvrščanje v skupine

Ta tehnika se uporablja za iskanje skupin (segmentov), v katere se uvrščajo objekti, glede na vrednost njihovih atributov. Objekti znotraj posamezne skupine imajo podobne lastnosti. Najlažje si razvrščanje v skupine predstavljamo na primeru množice oseb z dvema atributoma: starost in prihodek.



Slika 10: Osebe z atributoma *prihodki* in *starost* so uvrščene v 3 segmente

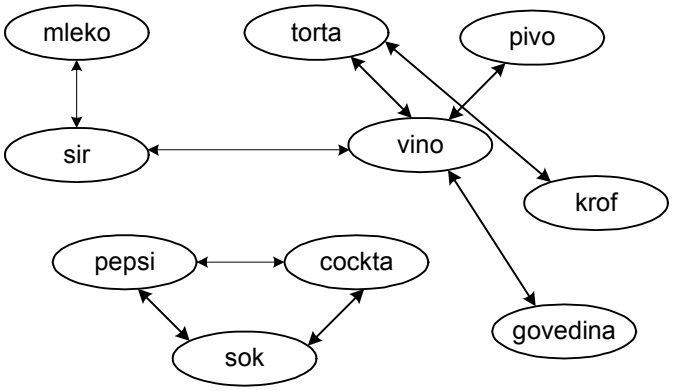
Osebe na zgornji sliki se delijo v tri segmente:

- Segment 1: mladina z nizkimi prihodki
- Segment 2: osebe srednjih let z višjimi prihodki
- Segment 3: upokojenci z relativno nizkimi prihodki

Razvrščanje v skupine uporabljamo v poslovni domeni za ciljno usmerjeno trženje, kjer nas zanima določena skupina subjektov, kot npr. zgoraj omenjeni segment mladine. V procesu podatkovnega rudarjenja to tehniko uporabljamo v začetni fazi za grobo analizo podatkov, kasneje pa za preverjanje uspešnosti npr. segmentnega trženja proti individualnem trženju.
Algoritem, ki določi skupine iz učne množice podatkov, spada v nenadzorovane. Vse atribute obravnava kot neodvisne.

5.1.3. Asociacije

Asociacijska pravila so se prvič uporabljala za analizo nakupovalne košarice. Na podlagi prodajnih transakcij lahko ugotovimo, kateri izdelki se v košarici pogosto pojavijo skupaj. Algoritem izračuna skupek pravil, ki jih lahko uporabljamo za zlaganje potencialno sorodnih izdelkov na isto polico.

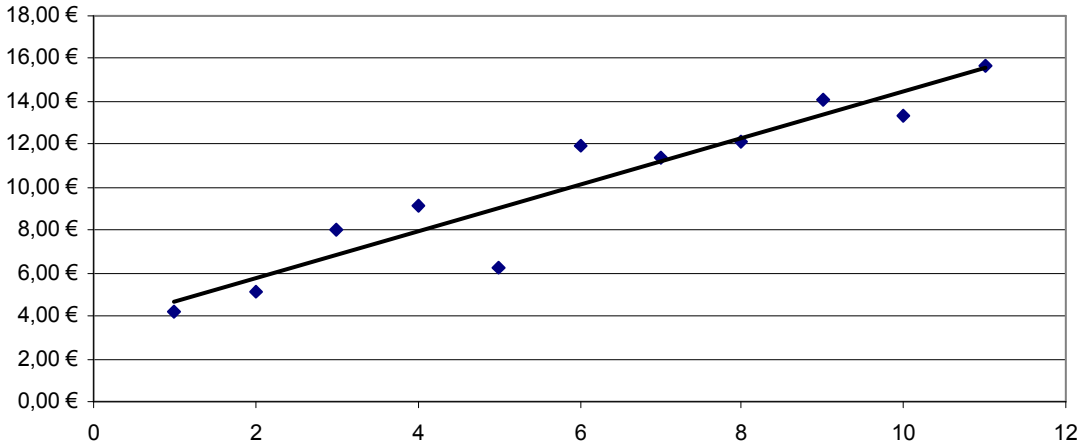


Slika 11: Primer povezav izdelkov v nakupovalni košarici

Asociacijska pravila se izračunajo po naslednjem postopku. Problem razdelimo na n učnih problemov, kjer je n število atributov (v zgornjem primeru je n kar število izdelkov). V podproblemu je izbran atribut odvisna spremenljivka, vsi preostali atributi pa so neodvisne spremenljivke. Z rešitvijo vsakega podproblema, izračunamo moč vpliva neodvisnih spremenljivk na odvisno in s tem povezave med njimi.

5.1.4. Regresija in napovedovanje

S to metodo, podobno kot pri klasifikaciji, določamo vrednost odvisne spremenljivke. Razlika je v tem, da je odvisna spremenljivka zvezna. Linearna regresija, kjer s premico aproksimiramo točke na koordinatnem sistemu, je tipičen primer uporabe regresije.



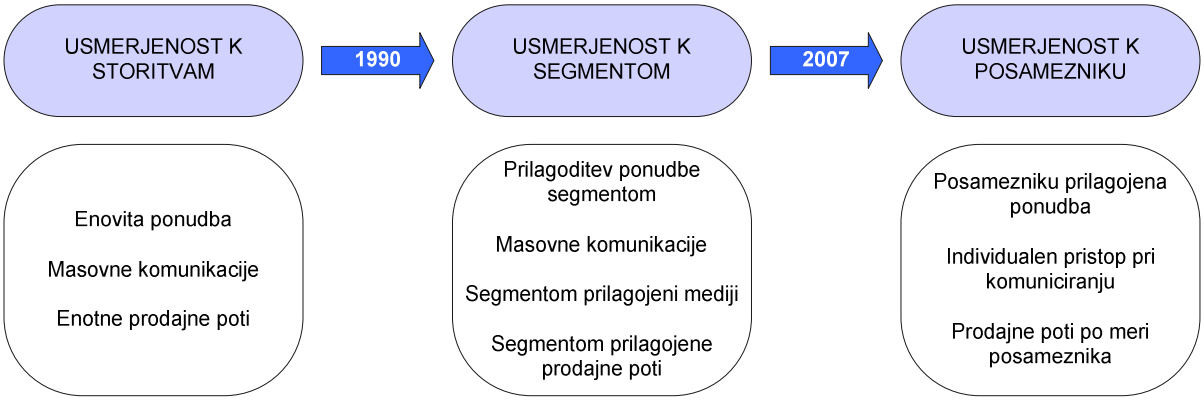
Slika 12: Gibanje tržne cene vrednostnega papirja v enem letu

Poznamo še logistično regresijo, kjer ciljna spremenljivka zavzema vrednost med 0 ali 1. Regresija se pogosto uporablja za reševanje poslovnih problemov. Primer je izračun beta koeficienta vrednostnega papirja na borzi ali npr. tržne cene vrednostnega papirja.

6. Izvedba postopka podatkovnega rudarjenja

6.1. Opis problema

Bančni sistemi, podobno kot vse organizirane družbe, sledijo trendom v informatiki. Velike organizacije se za spremembe, kot je vpeljava podatkovnega rudarjenja, odločijo po tem, ko je tehnologija že dobro vpeljana. Podobno je tudi v primeru banke, ki jo obravnavam v tej diplomski nalogi. Ker podatkovno rudarjenje uporabljam na področju trženja, se v nadaljevanju osredotočam na ta segment.



Slika 13: Dosedanji informacijski razvoj trženja v banki [13]

Razvidno je, da podatkovno rudarjenje spada v zadnjo razvojno fazo. Razlogi za vpeljano podatkovnega rudarjenja so predvsem v:

- večji natančnosti pri izbiri potencialnih strank
- hitrosti izvedbe projekta, saj z orodji RAD (ang. rapid application development) hitro pridemo do nekega rezultata
- merljivosti kvalitete, saj s individualnim pristopom dobimo ustrezen odziv stranke, letega pa lahko uporabimo za oceno kvalitete napovedi

Vsi našteti razlogi pomenijo za družbo nižje stroške trženja. Po uspešni izvedbi projekta podatkovnega rudarjenja in ob ustrezni integraciji v posameznih oddelkih, postane uporaba rezultatov del rutinskega dela zaposlencev. Masovna objava v javnih občilih je draga, poleg tega je odziv na akcijo slabo merljiv. Primer: Objava enkratnega oglasa, ki zaseda polovico notranje strani v časniku Delo, znaša v letu 2009: 7.915,60 EUR + DDV [14]. Za isto ceno bi lahko banka poklicala 1.000 najbolj potencialnih strank za sklenitev storitve.

V nadaljevanju je opisan proces izvedbe podatkovnega rudarjenja. Najprej so opisane posamezne faze po metodologiji CRISP-DM. Delo je potekalo na platformi SQL Server 2008. Za izdelavo modelov sem nato uporabil še orodje Weka. Na koncu sem zapisal primerjavo obeh orodij.

6.2. Izvedba procesa podatkovnega rudarjenja po metodologiji CRISP-DM

Funkcionalna razvejanost bančnega sistema in struktura informacijske tehnologije (IT) omogočata, da natančno sledimo fazam metodologije CRISP-DM. Industrijska praksa ocenjuje, da so potrebni časi za posamezne stopnje naslednji [11]:

- 50 do 70 odstotkov časa za pripravo podatkov
- 20 do 30 odstotkov časa za analizo podatkov
- 10 do 20 odstotkov časa za modeliranje, vrednotenje rezultatov in določitev poslovnih ciljev
- do 10 odstotkov časa za uporabo rezultatov

Tudi v obravnavanem primeru se je izkazalo, da ocene držijo. V nadaljevanju opisane faze ne vsebujejo vseh podrobnosti. Analiza stroškov in koristi bi denimo zahtevala celotno diplomsko nalogo, zato je ta del izpuščen.

6.2.1. Analiza poslovanja

V tej fazi je opisano stanje v banki pred uvedbo podatkovnega rudarjenja, kakšne cilje želimo doseči s tem projektom in izdelava projektnega načrta.

6.2.1.1 Določitev poslovnih ciljev

Cilj, ki ga želimo doseči s tem projektom, je dokazati uspešnost koncepta podatkovnega rudarjenja (ang. proof of concept). Za banko je dokaz koncepta višja natančnost trženja v primerjavi s klasičnimi metodami, kot sta segmentno usmerjeno ali masovno trženje. Kriterij uspešnosti je, da z uporabo tehnik podatkovnega rudarjenja dosežemo več odzivnih strank, kot bi jih z naključnim izborom ali določenim segmentom.

6.2.1.2 Ocena trenutnega stanja

Trenutno stanje opredeljujejo delovna sredstva in osebje, ki jih imamo na voljo za izvedbo projekta. Delovna sredstva predstavlja računalniška strojna in programska oprema. Delovna sredstva in osebje opisuje naslednja tabela.

Delovna sredstva	
Strojna oprema	Programska oprema
Strežnik z Windows Server 2008 (razvoj)	Podatkovno skladišče IBM DB2 (produkcija)
Strežnik z Windows Server 2008 (produkcija)	Podatkovno skladišče SQL Server 2008 (razvoj)
Delovna postaja Pentium 4, 1 GB RAM	Podatkovno skladišče SQL Server 2008 (testiranje)
Osebje	
Peter Konda, izvajalec projekta	
Tomaž Ogorevc, predstavnik oddelka za trženje	

Časovna usklajenost podatkov je pomemben dejavnik tveganja. Nepravilno upoštevanje pravil lahko pripelje do napačnega parametriziranja stranke, to pa do napačnih rezultatov podatkovnega rudarjenja. Takšen je npr. izračun strankinega mesečnega povprečja sredstev.

Če se izračun izvede npr. 15. v mesecu, je potrebno za pravilen izračun letnih povprečij to upoštevati. Takšnim tveganjem se izognemo tako, da v projekt vključimo poslovne in podatkovne analitike, ki dobro poznajo delovanje podatkovnih skladišč, v katerih se mesečno računajo razna povprečja.
Zaradi robustnosti sistema, velike količine zapisov in vhodnih atributov, lahko predpostavimo da je napaka statistično majhna. Zaradi hitrosti modeliranja sem število zapisov o strankah omejil na 5% aktivnih strank. Uporabljene stranke so fizične osebe s sklenjenimi storitvami.

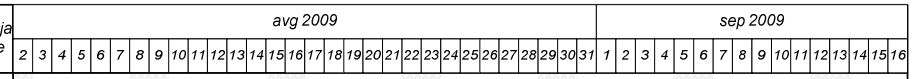
6.2.1.3 Cilji podatkovnega rudarjenja

Podatkovno rudarjenje v tem projektu bomo uporabili za napoved verjetnosti, ali bo stranka v določenem mesecu sklenila dolgoročni depozit. Dolgoročni depoziti imajo ročnost večjo od 12 mesecev. Natančnost rezultatov modela in s tem napovedi bo izmerjena na testnih podatkih s križnim preverjanjem (ang. cross validation).

6.2.1.4 Izdelava projektnega načrta

V projektnem načrtu podamo časovni plan za izvedbo projekta. Primer na sliki 14 prikazuje Ganttov diagram z opravili in časom trajanja posameznega opravila. Pri tem predpostavljamo, da bo projekt trajal od 1.8.2009 do 15.9.2009.

#	Opravilo	Začetek	Konec	Trajanje
1	Ocena poslovnega ozadja	3.8.2009	3.8.2009	1d
2	Ocena trenutnega stanja	4.8.2009	4.8.2009	1d
3	Določitev poslovnih ciljev	5.8.2009	6.8.2009	2d
4	Določitev ciljev podatkovnega rudarjenja, zapis projektnih tveganj in kriterijev uspešnosti	6.8.2009	7.8.2009	2d
5	Združitev podatkov iz DB2 v SQL Server 2008	10.8.2009	13.8.2009	4d
6	Opis podatkov in njihove strukture	12.8.2009	18.8.2009	5d
7	Izbor končnih atributov	17.8.2009	21.8.2009	5d
8	Izdelava podatkovne strukture	21.8.2009	21.8.2009	1d
9	Integracija zapisov	21.8.2009	26.8.2009	4d
10	Načrt modeliranja	27.8.2009	27.8.2009	1d
11	Modeliranje	28.8.2009	3.9.2009	5d
12	Ocenitev in izboljšava modelov	4.9.2009	10.9.2009	5d
13	Predstavitev rezultatov in ugotovitev	11.9.2009	11.9.2009	1d
14	Vrednotenje rezultatov	11.9.2009	14.9.2009	2d
15	Načrt uporabe rezultatov	15.9.2009	15.9.2009	1d



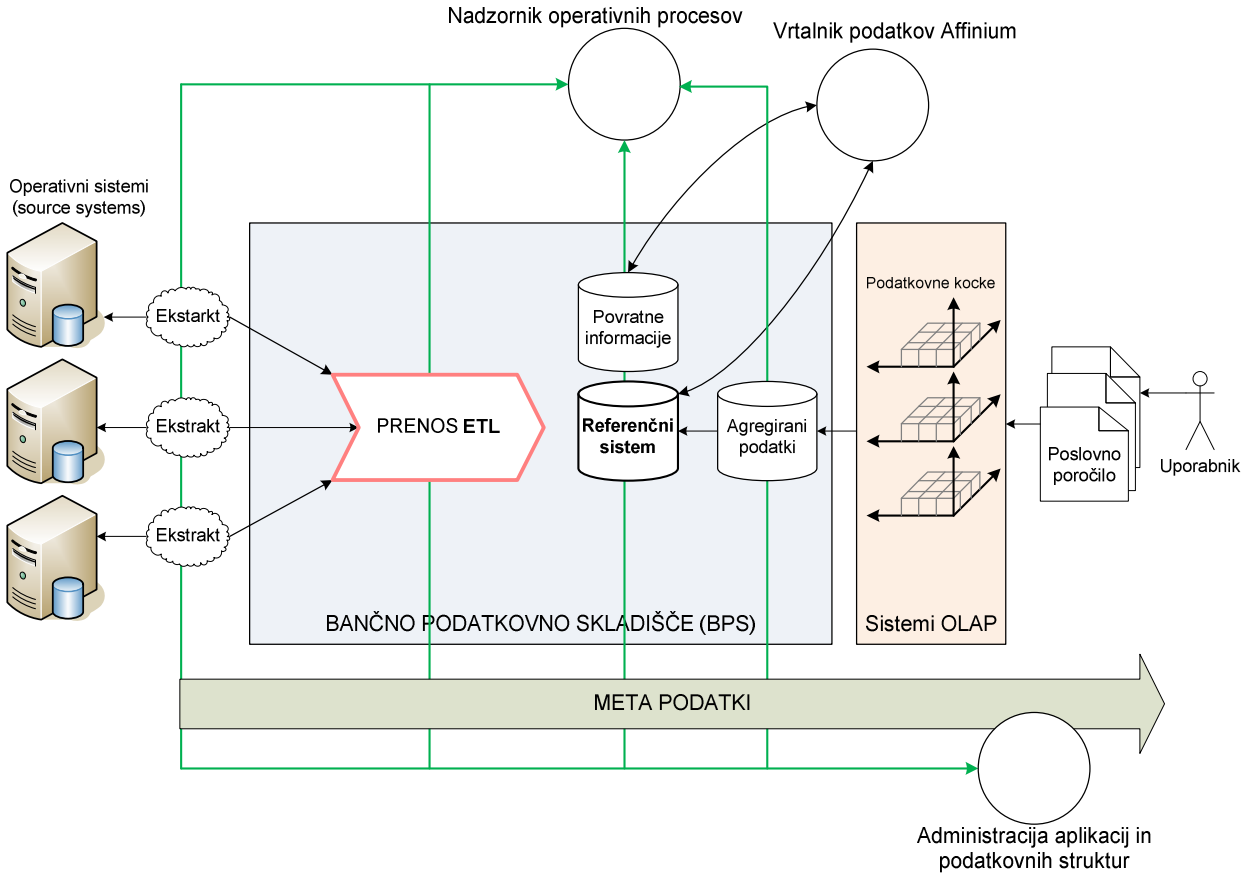
Slika 14: Ganttov diagram s podrobnostmi o poteku projekta

6.2.2. Analiza podatkov

V tej fazi se iz podatkovnega skladišča IBM DB2 prenesejo tabele, ki jih potrebujemo za izvedbo postopka. Opisati je potrebno strukturo teh tabel in izdelati seznam potencialnih atributov za podatkovno rudarjenje.

6.2.2.1 *Združevanje podatkov*

Tabele, ki se bodo uporabljale za podatkovno rudarjenje, je potrebno prenesti iz referenčnega podatkovnega skladišča IBM DB2 v razvojno okolje SQL Server 2008. Za prenos podatkov se uporablja orodje SQL Server Integration Services (SSIS). To je grafično orodje, ki omogoča prenose med različnimi podatkovnimi bazami. Preden se lotimo dejanskega prenosa, je potrebno dobro poznati obstoječo arhitekturo.



Slika 15: Arhitektura bančnega podatkovnega sistema

Iz operativnih sistemov je potrebno izdelati ekstrakte, ki se preko procesa priprave podatkov shranijo v repozitorij (IBM host). Prenosi ETL (ang. extract – transform – load) predstavljajo koncept polnjenja podatkovnega skladišča, ki vsebuje zaporedje operacij:

1. Ekstrakcija
2. Transformacija
3. Čiščenje in agregacija
4. Nalaganje

Prenosi ETL se izvajajo z uporabo orodij DTS (data transformation services) in novejšega SSIS, ki sta del sistema SQL Server. Za upravljanje prenosov ETL se uporablja sistem urnika (ang. scheduler), s katerim vodimo frekvenco transformacij.

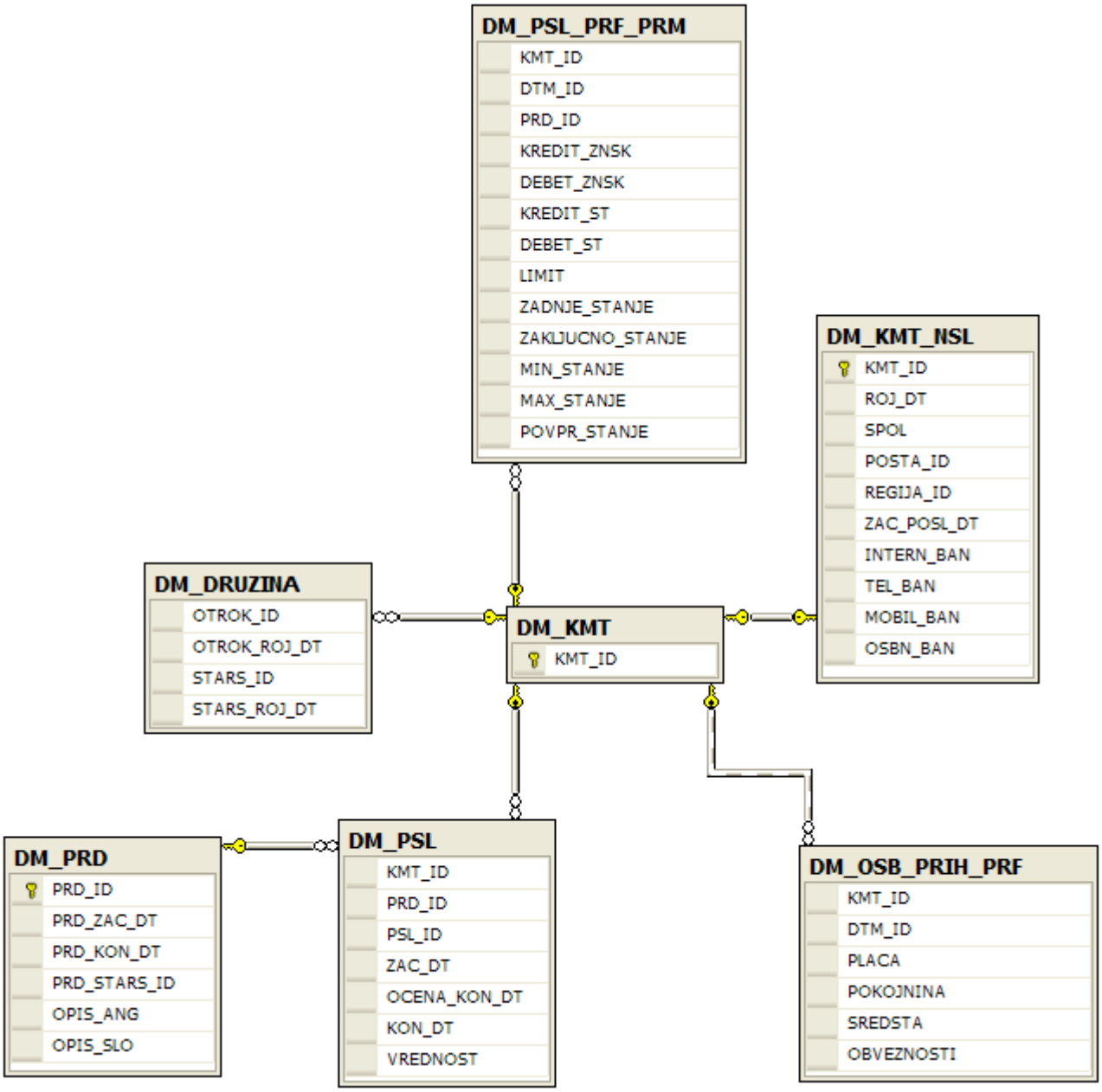
Aplikacije, ki se uporabljajo za evidenco in nadzor nad procesi ter pregled podatkov, so predstavljene s krogi. Kocke OLAP se uporabljajo za več-dimenzijski pogled na agregirane podatke.

6.2.2.1 Opis podatkov in njihove strukture

V naslednji tabeli je seznam uporabljenih tabel iz referenčnega sistema s kratkim opisom in periodo polnjenja.

Ime tabele	Dolgo ime	Opis	Frekvenca polnjenja
KMT	Komitent	Komitent je z banko povezana oseba, ki z njo nujno ne posluje.	dnevno
KMT_PRF	Poslovanje komitenta	Stanje komitenta vsebuje atribute, kot so: kredit, debet, število kreditnih in debetnih transakcij, stanje sredstev in obveznosti, število opravljenih storitev, itd.	mesečno
LOK	Lokacija	Lokacijo sestavljajo atributi: regija, mesto, občina.	dnevno
NSL	Naslov komitenta	Naslov predstavlja določeno vrsto lokacije povezane osebe za namene pošiljanja in sprejemanja pomembnih informacij.	dnevno
OSB	Fizična oseba	Fizična oseba je podtip komitenta. To je katerakoli fizična oseba, ki je prišla v stik z banko in o kateri ima banka v svojem registru podatke. Fizična oseba ni nujno tudi stranka banke.	dnevno
OSB_PRIH	Prihodki fizične osebe	Prihodki na račun fizične osebe, kot so plača in pokojnina. Vsebuje tudi celotni prihodek.	mesečno
PRD	Storitev (produkt)	Produkt identificira storitve, ki jih banka ponuja in prodaja. Storitev je ključna dimenzija podatkovnega skladišča.	dnevno
PSL	Posel	Posel predstavlja vsako veljavno sklenjeno pogodbo med banko in komitentom.	dnevno
PSL_PRF	Donosnost posla	Donosnost na nivoju posameznega posla.	mesečno
PSL_PRM	Promet posla	Promet po poslih. Entiteta ne vsebuje samo prometa, temveč tudi druge pomembne atribute. Npr.: Minimalno stanje, maksimalno stanje, število transakcij itd.	mesečno
RLC	Relacije	Uporablja se pri izračunu števila otrok stranke.	mesečno
VP_PSL	Trgovanje z vrednostnimi papirji	Promet z vrednostnimi papirji za tuji račun. Entiteta ne vsebuje samo prometa, temveč tudi druge pomembne atribute. Primer.: število transakcij, opravnine, itd.	mesečno

Tabele sem izbral po sestanku s predstavnikom oddelka za trženje. Dejstvo, da opisane tabele prenašamo iz podatkovnega skladišča, olajšuje delo pri analizi podatkov, saj so podatki že primerno združeni na dnevnem/mesečnem nivoju.
Izbrane tabele nato z orodjem SSIS prenesemo v SQL Server 2008 razvojno okolje. Tabele, ki jih dobimo s prenosom, vsebujejo samo relevantne podatke za podatkovno rudarjenje. Povezave med tabelami prikazuje entitetno-relacijski model na sliki 16.



Slika 16: Združene tabele iz referenčnega sistema, pripravljene za analizo

Naslednja tabela prikazuje podrobnosti tabele OSB_PRIH, to so prihodki osebe na mesečnem nivoju. Podrobnosti ostalih tabel so izpuščene.

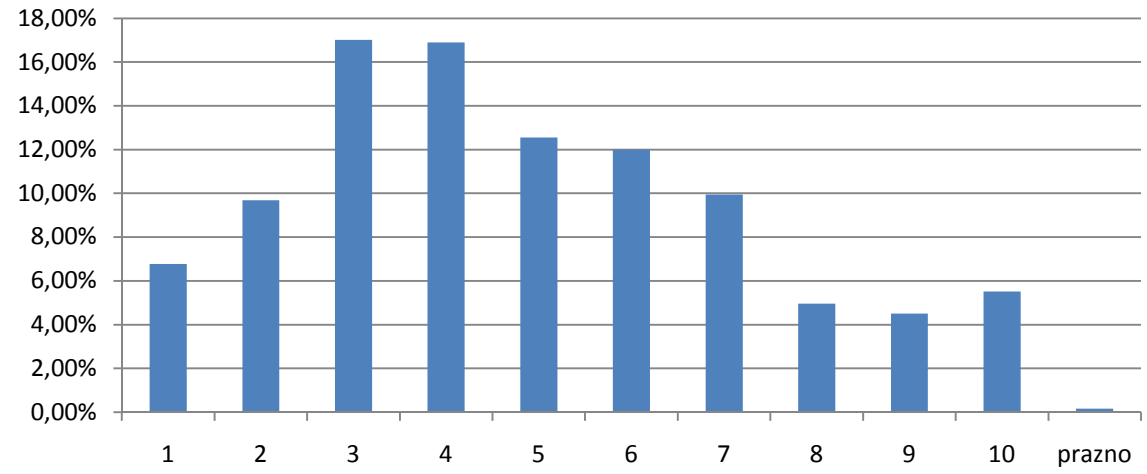
Atribut	Tip	Dolžina	Opis
DTM_ID	SMALLINT	10	Identifikacija datuma za mesečni nivo. Primer: 908 (avgust 2009)
PSL_ID	INTEGER	5	Identifikacija posla
KMT_ID	INTEGER	5	Identifikacija komitenta
PRD_ID	SMALLINT	5	Identifikacija produkta (storitve)
OE_ID	SMALLINT	5	Identifikacija organizacijske enote
PLACA	DECIMAL	12	Plača
POKOJNINA	DECIMAL	11	Pokojnina
OSB_OST_PRIH	DECIMAL	11	Ostali prihodki na računu
OSB_CEL_PRIH	DECIMAL	11	Celotni prihodki na računu

PRENOS_DT	DATE	9	Datum prenosa iz transakcijske baze v podatkovno skladišče
OSB_NET_STN	DECIMAL	11	Neto znesek na računu

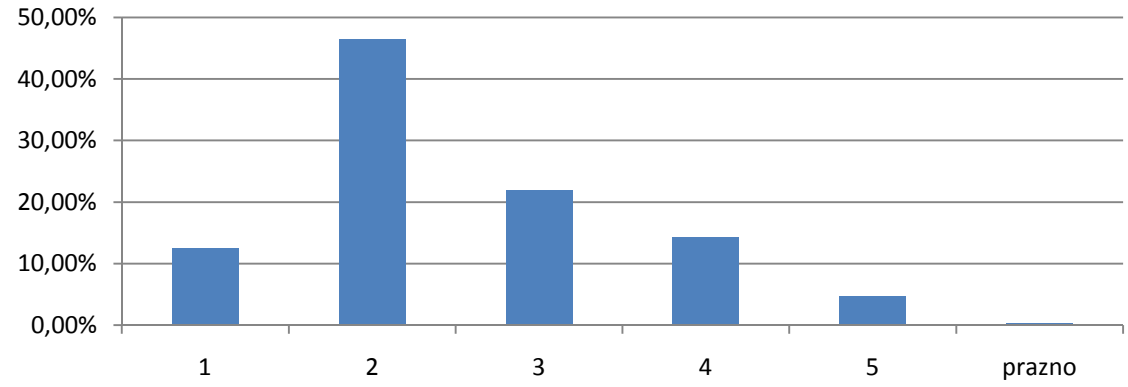
6.2.2.2 Podrobna raziskava podatkov

Pri raziskovanju podatkov nas zanima predvsem frekvenčna porazdelitev strank po vrednosti atributov. V nadaljevanju prikazujem frekvenčne porazdelitve nekaterih pomembnih atributov.

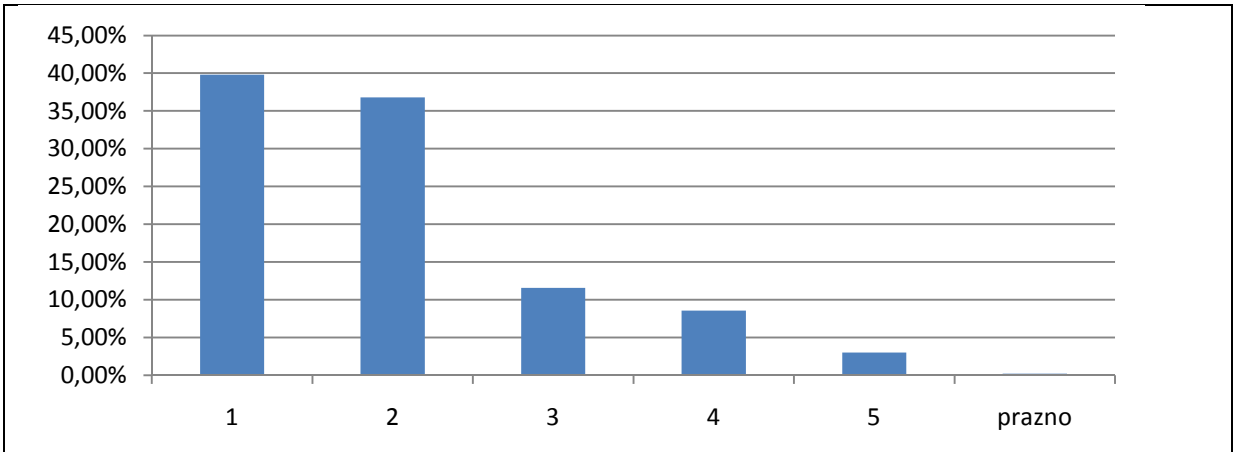
Segment prihodkov (plača, pokojnina) – višja vrednost pomeni višji prihodek



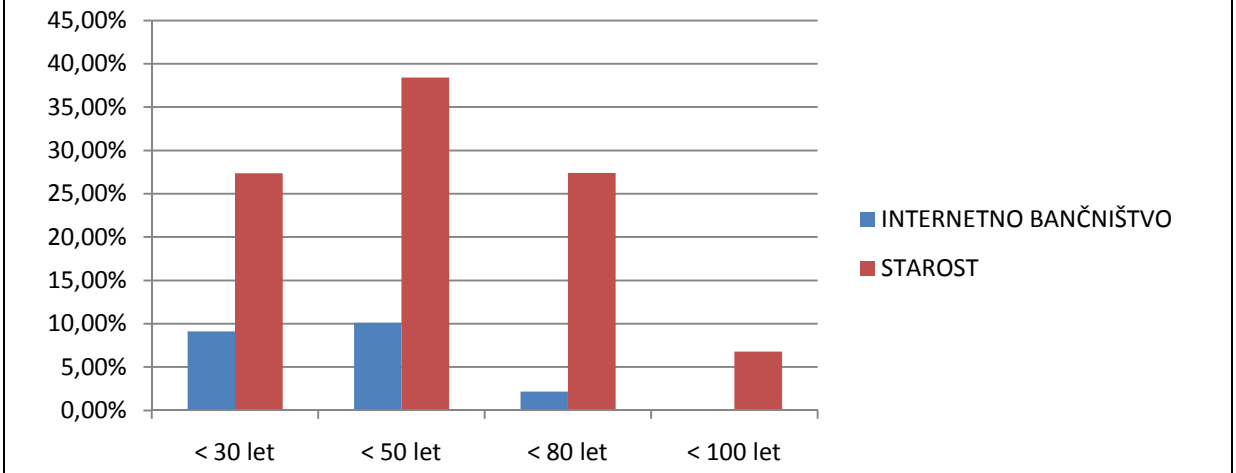
Segment sredstev – višja vrednost pomeni več sredstev



Segment obveznosti – nižja vrednost pomeni manjšo zadolženost



Razmerje med starostjo strank in uporabo internetnega bančništva



6.2.2.3 Ocena kakovosti podatkov

Kakovost podatkov sem ocenjeval skozi posamezne iteracije preko rezultatov modelov. Izpuščeni atributi so opisani v poglavju 6.2.3.3.

6.2.3. Priprava podatkov

Rezultat te faze je podatkovna zbirka, ki se bo uporabljala za izdelavo modela.

6.2.3.1 Izbor podatkov

Izbrati moramo podatke za izdelavo modela. To pomeni, da izberemo konkretne primere oz. množico zapisov (ang. rows), ki bodo sodelovali pri učenju modela in atribute teh primerov. Atribute predstavljajo stolpci v tabeli (ang. columns). Pravila, ki omejujejo množico komitentov so naslednja:

- Fizične osebe (1:1 povezava tabel KMT in OSB)
- Aktivne osebe (število sklenjenih poslov ≥ 0)
- Starost fizične osebe med 18 in 95 let
- 10% vseh oseb (ID komitenta *modulo* $20 \leq 1$)

Prvotni nabor pravil je vseboval le aktivne fizične osebe v obsegu 5,5%. V naslednjih iteracijah se je izkazalo, da je smiselno omejiti stranke po letih ter vzeti večji nabor.

Začetni nabor atributov za modeliranje sem izbral na podlagi podatkov oddelka za trženje. Ta že uporablja segmentacijo za izračun relativnega položaja stranke (bonitetna ocena). Za izračun uporabljajo izpeljane (agregirane) atribute, ki bi lahko bili primerni tudi za modeliranje. Izbrane atribute prikazuje naslednja tabela.

Atribut	Tip	Opis
KMT_ID	INT	Identifikator fizične osebe
PSL_ID	INT	Identifikator sklenjene storitve
PSL_ZAC_DT	DATETIME	Datum začetka veljavnosti storitve PSL_ID
PRD_STARS_ID	INT	Identifikacija skupine produktov iz šifranta produktov; s tem atributom lahko zajamemo vse dolgoročne depozite, ki nas v obravnavi zanimajo.
DTM_ID	INT	Identifikacija datuma (mesec, leto)
ZAC_POSL_DT	DATETIME	Datum priključitve fizične osebe v banki
INTERN_BAN	SMALLINT	{0,1}; Oseba uporablja internetno bančništvo
MOBIL_BAN	SMALLINT	{0,1}; Oseba uporablja mobilno bančništvo
OSBN_BAN	SMALLINT	{0,1}; Oseba ima osebnega referenta
TEL_BAN	SMALLINT	{0,1}; Oseba uporablja telefonsko bančništvo
REGIJA_ID	INT	Identifikator regije, v kateri oseba stanuje
ROJ_DT	DATETIM	Datum rojstva osebe
POKOJNINA	DECIMAL	Pokojnina
PLACA	DECIMAL	Plača
SREDSTVA_POVPR	DECIMAL	Sredstva, npr. depoziti, varčevanja, delnice, itd.
OBVEZNOSTI_POVPR	DECIMAL	Obveznosti, npr. krediti, negativno stanje, kreditne kartice, ipd.
OTROK_ST	INT	Število otrok
MIN_STANJE	DECIMAL	Minimalno stanje
POVPR_STANJE	DECIMAL	Povprečno stanje
KRED_STANJE	DECIMAL	Povprečje zneska kreditnih transakcij
RAC_NET_STANJE	DECIMAL	Neto vrednost na računu

Atributi so se skozi iteracije močno spreminjali, kar je razvidno v naslednjih korakih procesa. Do sprememb ali odstranitev atributov je prišlo zaradi redkosti vhodnih podatkov (ang. sparse data), ali zaradi izpeljave podatkovnega tipa. Predvsem zvezni atributi, npr. plača ali sredstva, so bili v kasnejših iteracijah zamenjani z ustreznimi tržnimi segmenti. Razlog je v tem, da notranja diskretizacija zveznih atributov, ki jo uporabljajo algoritmi v orodju Analysis Services, ni tako uspešna, kot uporaba lastnih segmentnih vrednosti. Bančni segmenti so fino prilagojeni slovenskemu makroekonomskemu stanju, kar bistveno pripomore h kvaliteti modela.

6.2.3.2 Čiščenje podatkov

Uporaba podatkovnega skladišča, pri katerem dobimo preračunane vrednosti, pomeni, da ne potrebujemo dodatne analize za čiščenje podatkov. Čiščenje namreč že izvajamo na dveh (nižjih) nivojih v procesu prenosa podatkov:

1. Pri pripravi dela s prenosom iz operativnih sistemov v referenčno podatkovno skladišče.
2. Pri prenosu iz referenčnega v agregirano področje.

6.2.3.3 Izdelava podatkovne strukture

Identifikacija izpeljanih atributov praviloma poteka v več iteracijah. Transformacijo iz začetnih atributov v končne prikazuje naslednja tabela.

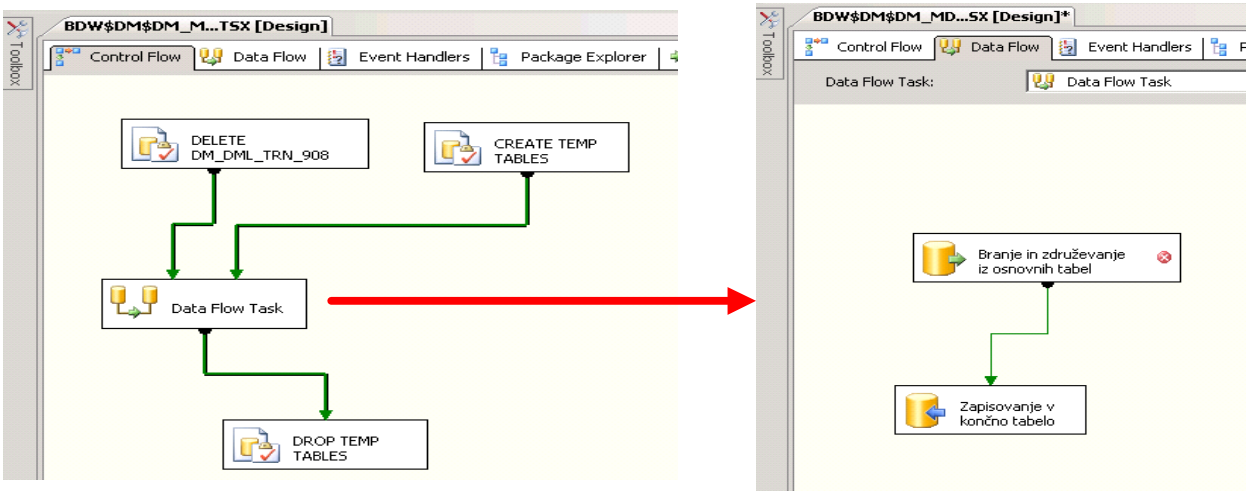
Prva iteracija	Končna iteracija	Opis	Obdobje
KMT_ID	KMT_ID	Identifikator fizične osebe	908 (avg. 2009)
DTM_ID	(ni uporabljeno)	V koraku integracije časovne atribute povprečimo	
REGIJA_ID	CITY_BIG	{0,1}; Veliko mesto, npr. Ljubljana, Celje, itd.	908
	REGIJA_OSREDNJA	{0,1}; Osrednja regija	
ROJ_DT	AGE_CLASS	{1,2,3,4,5, 99}; Starostni razred osebe	908
POKOJNINA	SLRY_PENS_SEG_ID	{0..10}; Segment osebnega dohodka izračunamo 1 mesec pred izračunom T	907
PLACA			
SREDSTVA_POVPR	TOT_AST_SEG_ID	{1..5}; Segment sredstev	808-907
OBVEZNOSTI_POVPR	TOT_LBY_SEG_ID	{1..5}; Segment obveznosti	808-907
OTROK_ST	(ni uporabljeno)	Premalo podatkov o strankah z otroci, zato atribut ni primeren za modeliranje	908
MIN_STANJE	MN_BAL_F	{0,1}; Minimalno stanje 0 = negativno stanje, 1 = pozitivno stanje	808-907
POVPR_STANJE	AV_BAL_F	{0,1}; Povprečno stanje 0 = negativno stanje, 1 = pozitivno stanje	808-907
KRED_STANJE	CT_AMT_F	{0,1}; Povprečje zneska kreditnih transakcij 0 = brez transakcij, 1 = pozitiven znesek	808-907
RAC_NET_STANJE	ACC_NET_F	{0,1}; Neto vrednost na računu 0 = negativna vrednost, 1 = pozitivna vrednost	808-907
INTERN_BAN	INTERN_BAN	{0,1}; Internetno bančništvo (Klik)	908
MOBIL_BAN	MOBIL_BAN	{0,1}; Mobilno bančništvo	908
OSBN_BAN	OSBN_BAN	{0,1}; Osebno bančništvo	908
TEL_BAN	TEL_BAN	{0,1}; Telefonsko bančništvo (Teledom)	908
Novi (izpeljani) atributi			
	TRG	{0,1}; Razred cilj – oseba sklenila dolgoročni depozit v izbranem mesecu	908
	T_END	{0, 1}; Potek depozita v določenem mesecu. Atribut ni bil uporabljen zaradi premočnega vpliva na model.	908
	T_LY	Število sklenjenih depozitov v obdobju pred TRG	808-907
	MAX_DATE_DEP	Število mesecev od zadnjega sklenjenega dolgoročnega depozita	pred 908
	MAX_DATE_WITHOUT_DEP	Število mesecev od zadnjega poteklega dolgoročnega depozita	pred 908
	MAX_DATE_ALL	Število mesecev od zadnje sklenjene storitve	pred 908
	AR_CNT_SEG_ID	{1..5}; Segment števila vseh sklenjenih storitev	808-907
	CST_RLN_AGE_SEG_ID	{1..5}; Segment dolžine sodelovanja z banko	808-907

Atributi, predstavljeni z mesečnim obdobjem, so v resnici posnetki stanja v tem mesecu in ne povprečja. Izpeljan atribut TRG je poudarjen, ker predstavlja odvisno spremenljivko, ki se uporablja za izračun pravil. Pod (TRG = 1) spadajo vsi depoziti z ročnostjo večjo od enega leta.

6.2.3.4 Integracija zapisov

Tabele iz slike 16 z uporabo poizvedbe SQL združimo v eno tabelo, ki bo uporabljena v fazi modeliranja. Integracijo v končno tabelo sem izvedel v obliki projekta SSIS, ki ga prikazuje zaslonska slika 17.

V kontrolnem toku (ang. control flow) se najprej izvedeta dve poizvedbi: brisanje končne tabele DM_MDL_TRN_908 in izdelava začasnih tabel za hitrejše delovanje. Podatkovni tok (ang. data flow) prikazuje prenos podatkov iz osnovnih tabel v končno tabelo. Na koncu se v kontrolnem toku pobrišejo začasne tabele.



Slika 17: Izdelava končne tabele za modeliranje

6.2.3.5 Formatiranje vrednosti

Formatiranje vrednosti atributov je sintaktično in naj ne bi spreminjalo pomena podatkov. Komponenta za modeliranje Analysis Services, ki je del platforme SQL Server 2008, je zmožna upravljati z različnimi tipi podatkov. Omogoča tudi samodejno pretvorbo zveznih podatkov tipov v diskretne. Zaradi možnosti samodejne pretvorbe in dejstva, da tekstovna polja niso bila uporabljena, eksplicitna pretvorba podatkov ni bila potrebna.

6.2.4. Modeliranje

Cilj podatkovnega rudarjenja je izračunati verjetnosti sklenitve depozita za mesec september 2009. Ali stranka sklene depozit ali ne, določa razredni atribut TRG – cilj (ang. target). TRG je diskreten atribut tipa *boolean*, ki je lahko 0 ali 1. Za učenje modela sem uporabil izbrane podatke strank za mesec avgust 2009.

6.2.4.1 Izbira tehnik

Za modeliranje sem izbral 4 nadzorovane klasifikacijske algoritme: odločitvena drevesa, nevronske mreže, logistična regresija in naivni Bayesov klasifikator. Pri vseh uporabljenih algoritmih sem pustil privzete nastavitve parametrov. Orodje Analysis Services na podlagi števila vhodnih atributov in njihovih podatkovnih tipov samo ugotovi, kakšne so optimalne nastavitve. Za primer navajam parametre algoritma Microsoft decission trees, ki so opisane v naslednji tabeli.

Parameter	Vrednost	Opis
COMPLEXITY_PENALTY	[0, 1]	Določa razvejanost drevesa. Višja vrednost pomeni, da bo algoritem močneje rezal veje drevesa. Algoritem uporablja tehniko sprotnega rezanja (ang. forward prunning)
MINIMUM_SUPPORT	celo število	Minimalno število zapisov, ki jih mora vsebovati vozlišče drevesa, da cepitev drevesa poteka naprej
SCORE_METHOD	{1,2,3}	Analitična metoda, po kateri se izbere vozlišče za cepitev veje: 1 – entropija 2 – algoritem K2 za učenje iz Bayesovih mrež 3 – algoritem BDEU
SPLIT_METHOD	{1,2,3}	Metoda cepljenja vozlišča: 1 – binarna 2 – cepitev po vseh stanjih 3 – samodejna izbira med 1. in 2.
MAXIMUM_INPUT_ATTRIBUTES	celo število	Če je parameter podan, algoritem upošteva omejeno najboljših vhodnih atributov, ostale pa ignorira
MAXIMUM_OUTPUT_ATTRIBUTES	celo število	Enako kot zgoraj, vendar za izhodne atribute
FORCE_REGRESSOR	niz	Pri napovedi zveznih atributov s tem parametrom izberemo vhodne atribute, po katerem se vozlišča cepijo v veje

6.2.4.2 Načrt modeliranja

Preden se lotimo izdelave modela, je potrebno izbrati učno in testno množico podatkov. Prva se uporablja za indukcijo pravil modela, druga za testiranje naučenih pravil. Na prvi pogled se zdi, da obravnavani primer potrebuje specifično obravnavo, ker razmerje odzivnih strank proti neodzivnim znaša približno 1:25, torej manj kot 5%. To lahko, podobno kot pri npr. problemu pranja denarja, predstavlja težavo pri izdelavi modela [15]. Algoritem lahko izpusti pomembna pravila ali pa jih preveč prilagodi učni množici. To se je izkazalo v prvih iteracijah projekta, ko je bila napovedna moč modela slaba. Problem rešujemo tako, da število pozitivnih (razred TRG = 1) zapisov prilagodimo s povečevanjem (ang. overfitting) ali zmanjševanjem (ang. underfitting). Drugi način reševanja je, da v samem vzorcu uporabimo razvrščanje v skupine, za učenje pa nato uporabimo samo določene skupine.

V prvih iteracijah se je prilagoditev učne množice izkazala za uspešno metodo izboljšanja modelov. Vzorec za učno množico sem omejil z naslednjim pogojem:
(TRG=1) OR (TRG=0 AND KMT_ID % X = 1)

Upošteval sem vse odzivne stranke, neodzivne pa sem naključno omejil. S to omejitvijo sem razmerje odzivnih strank proti neodzivnim popravil na 1:4.
Pri končni podatkovni strukturi, kjer so bile vrednosti atributov zamenjane z ustreznimi segmenti, je prilagojena učna množica z razmerjem 1:4 bistveno poslabšala rezultat. Te ugotovitve so tudi merljive. Naj bo matrika razvrstitev (ang. confusion matrix):

	0 (dejanska vrednost)	1 (dejanska vrednost)
0 (napovedana vrednost)	TN	FN
1 (napovedana vrednost)	FP	TP

Pri tem so oznake:
TN – število pravilno klasificiranih negativnih primerov
FN – število nepravilno klasificiranih negativnih primerov (napaka tipa II)
FP – število nepravilno klasificiranih pozitivnih primerov (napaka tipa I)
TP – število pravilno klasificiranih pozitivnih primerov

Sedaj lahko izračunamo naslednje statistične ocene:

$$A = \frac{TN + TP}{TN + TF + FP + TP}$$ Accuracy (natančnost)

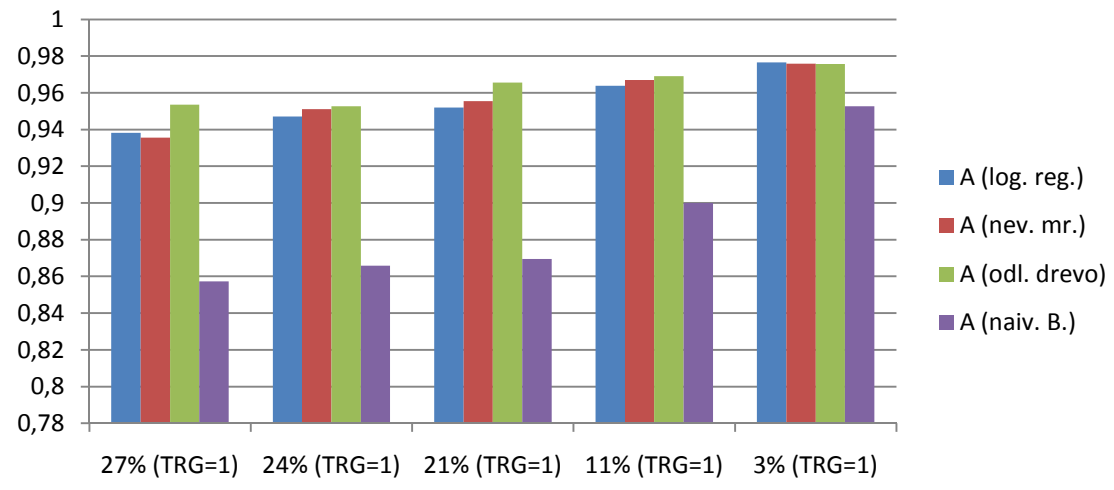
$$P = \frac{TP}{TP + FP}$$ Precision (točnost)

$$R = \frac{TP}{TP + FN}$$ Recall (občutljivost)

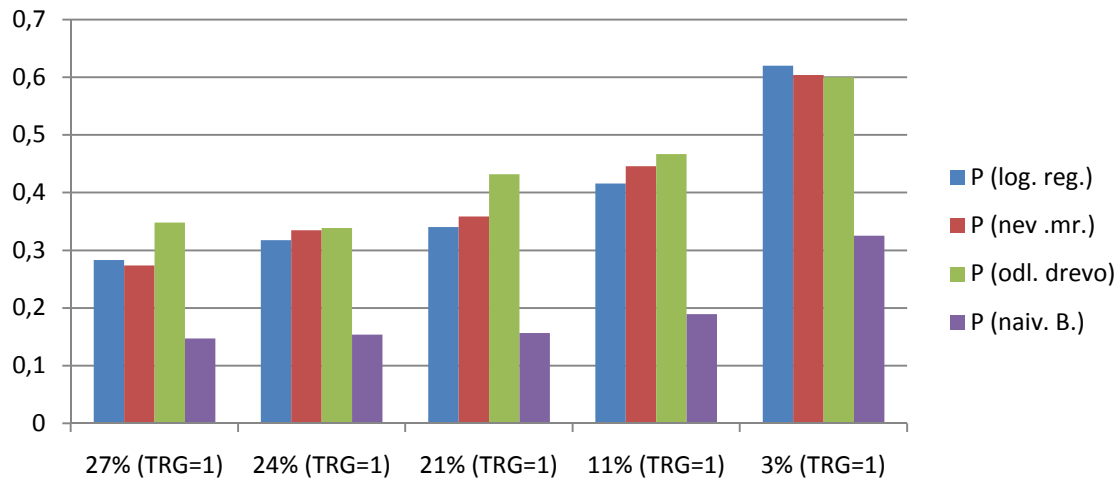
$$F_\beta = (1 + \beta^2)\frac{P * R}{(\beta^2) * P + R}$$ F-measure (harmonična sredina med točnostjo in občutljivostjo)

Spodnji grafi prikazujejo rezultate na testnih podatkih (n=80.202). Vodoravna os kaže velikost odzivnih strank v vzorcu, pri čemer stolpci na skrajni desni predstavljajo neprilagojen vzorec s 3% odzivnih strank.

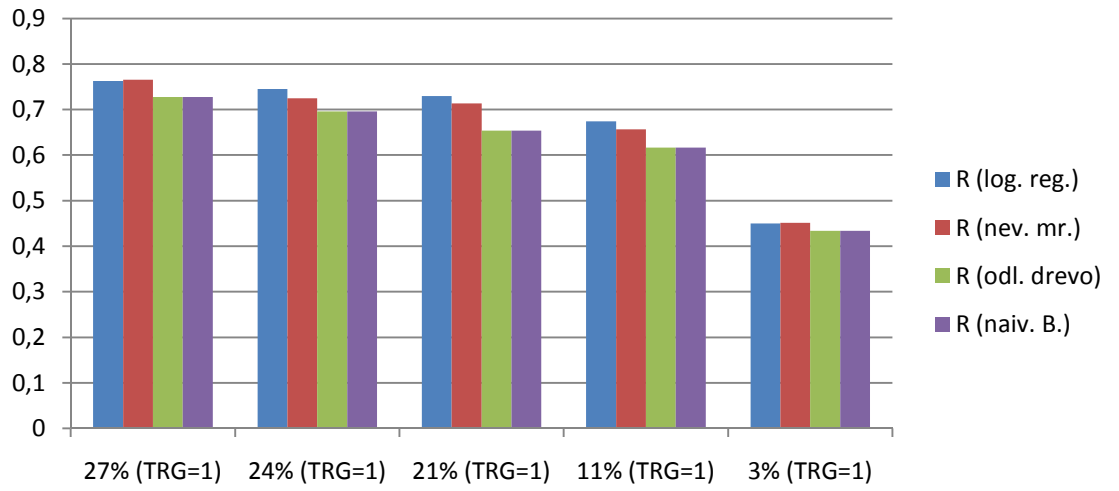
Accuracy (natančnost):



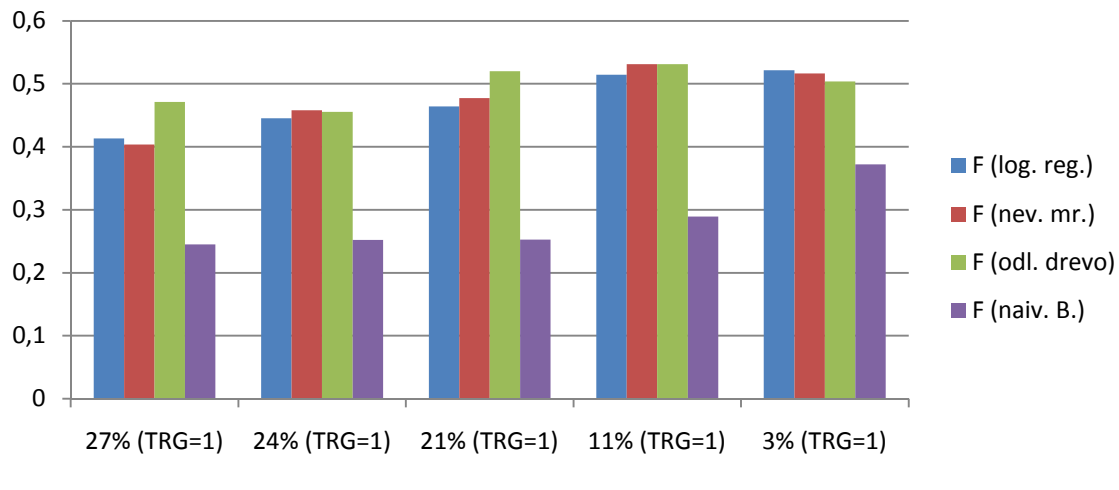
Precision (točnost):



Recall (občutljivost):



F-measure (harmonično razmerje med točnostjo in občutljivostjo):



Z večanjem odzivnih strank v vzorcu raste točnost in pada občutljivost, njuna harmonična sredina pa ostaja približno enaka od razmerja 1:10 in navzgor. Kljub sicer dobrim obetom pri vzorcu z 11% odzivnih strank, sem se odločil, da za model uporabim nespremenjeno množico

podatkov. Razlog je v tem, da se je logistična regresija izkazal kot najboljši algoritem na novih podatkih in ta algoritem ima najboljše razmerje med točnostjo in odzivnostjo ravno pri nespremenjeni učni množici. Drugi razlog je v tem, da ima nespremenjena množica najboljšo natančnost (A) pri vseh algoritmih, kar pomeni, da bodo ti modeli v splošnem napovedovali natančneje kot modeli, naučeni z zelo prilagojeno učno množico.

Za testiranje modelov sem uporabil drugačen nabor strank s podatki za mesec september 2009. Testiranje na drugačnem naboru je potrdilo, da pravila niso prilagojena samo za učno množico. Točnost napovedovanja sem preveril z:

- uporabo odzivnega diagrama (ang. lift chart), ki je izpeljava krivulje ROC (ang. receiver operating characteristic) [16]
- uporabo K-kratnega križnega preverjanja (ang. K-fold cross-validation), kjer merimo število pravilno in nepravilno klasificiranih primerov, logaritemsko oceno, odziv in kvadratni koren povprečne kvadratne napake

6.2.4.3 Modeliranje

Pred dejansko izdelavo modela smo preučili in pripravili podatke za izdelavo modela, izbrali ustrezne tehnike modeliranja in tehnike za validacijo rezultatov. Sledi izdelava modelov z izbranim orodjem. V tem primeru uporabljamo razvojni uporabniški vmesnik za Visual Studio. Program Analysis Services za izdelavo modelov teče kot strežni proces. Namesto uporabniškega vmesnika lahko uporabimo jeziku SQL (ang. structured query language) podoben strukturni jezik DMX (ang. data mining extensions). Oba pristopa za izdelavo modela sta funkcionalno identična. Proces izdelave modelov je naslednji:

1. Določimo izvor podatkov, ki je lahko podatkovna baza ali kocka OLAP. Izberemo podatkovno bazo DM na razvojnem okolju SQL Server 2008
2. Določimo tabelo, na kateri se bo izvajalo podatkovno rudarjenje. Izberemo tabelo za učenje, ki smo jo definirali v koraku Načrtovanje modeliranja. Orodje podpira tudi ugnezdene tabele, kar je koristno npr. pri asociacijskih pravilih, v našem primeru pa ugnezdenih tabel ne potrebujemo.
3. Izberemo tehnike za podatkovno rudarjenje. V našem primeru so tehnike naslednje: Microsoft Decission Trees, Microsoft Logistig Regression, Microsoft Neural Network in Microsoft Naive Bayes.
4. Izberemo vhodne in izhodne atribute ter ključe. Naslednja tabela vsebuje prikazuje vse atribute za posamezne algoritme in njihove tipe.

Atribut	Tip	Logistična regresija	Nevronske mreže	Odločitvena drevesa	Naivni Bayes
ACC NET AMT F	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
AGE CLASS	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
AR CNT SEG ID	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
AV BAL F	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
CITY BIG	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
CST RLN AGE SEG ID	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
CT AMT F	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
INTERN BAN	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut

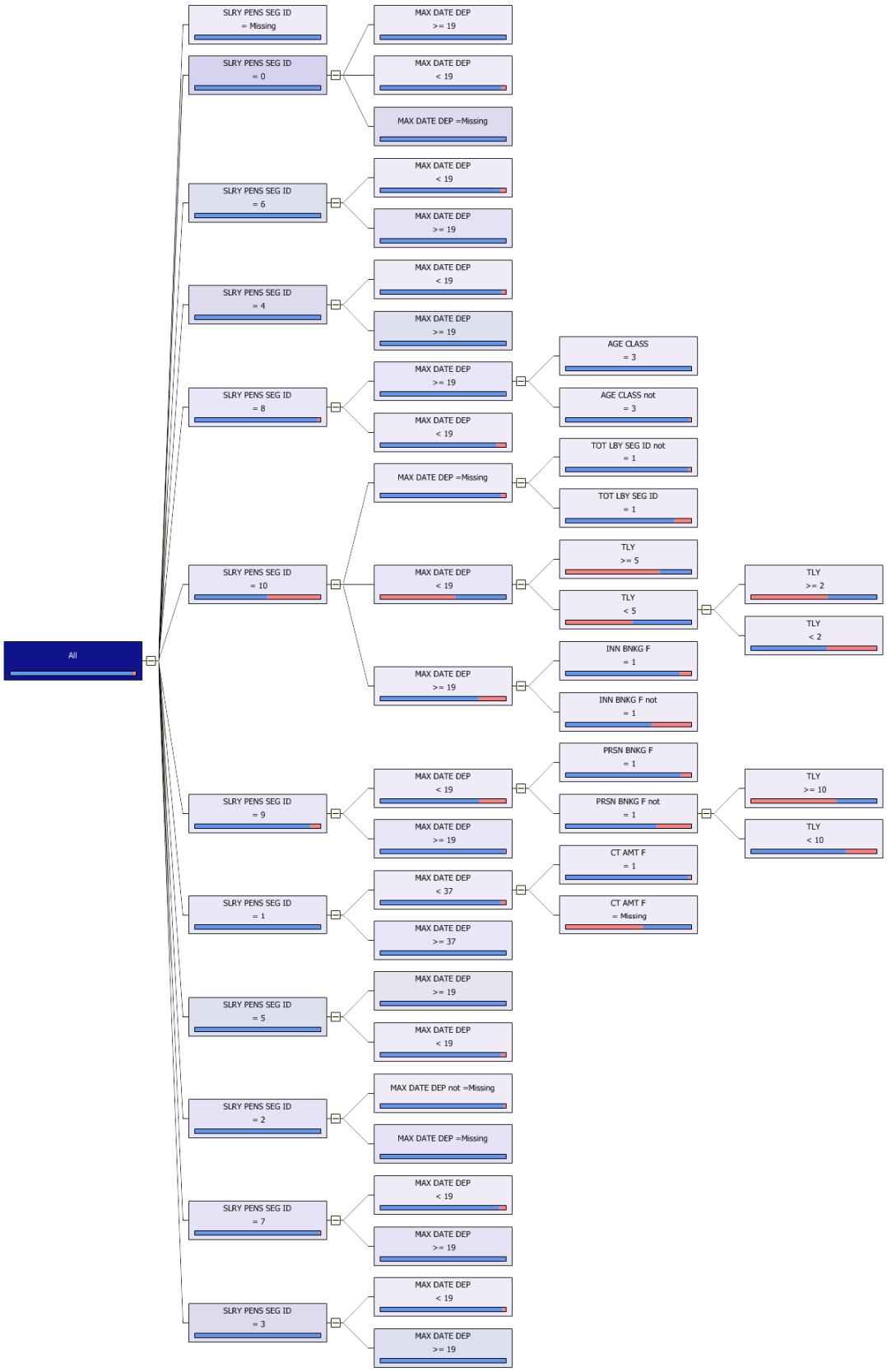
KMT ID	Zvezni	Identifikator	Identifikator	Identifikator	Identifikator
MAX DATE ALL	Zvezni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Ignoriraj
MAX DATE DEP	Zvezni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Ignoriraj
MAX DATE WITHOUT DEP	Zvezni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Ignoriraj
MN BAL F	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
MOBIL_BAN	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
OSBN_BAN	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
REGIJA OSREDNJA	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
SLRY PENS SEG ID	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
TEL_BAN	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
TLY	Zvezni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Ignoriraj
TOT AST SEG ID	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
TOT LBY SEG ID	Diskretni	Vhodni atribut	Vhodni atribut	Vhodni atribut	Vhodni atribut
TRG	Diskretni	Razred	Razred	Razred	Razred

Algoritem naivni Bayes atribute T_LY, MAX_DATE_ALL, MAX_DATE_DEP, MAX_DATE_WITHOTUT_DEP in MAX_DATE_ALL ignorira, ker so zvezni. Ostali algoritmi znajo pravilno delovati tudi z zveznimi atributi. Za vse naštete atribute lahko izberemo diskretizacijo, vendar so bili v tem primeru modeli za 10% slabši.

5. Izberemo odstotek zapisov, ki ga program vzame na stran za testiranje. Za učenje pravil internega testiranja ne potrebujemo. Vhodna tabela je že prilagojena za učenje, zato ta parameter postavimo na 0%, validacijo pa kasneje izvajamo na ločeni tabeli.
6. Nastavimo parametre za posamezne tehnike. Orodje samo ugotovi, kakšne so najboljše nastavitve glede na količino vhodnih atributov in števila zapisov, zato sem ohranil privzete nastavitve.
7. Zaženemo procesiranje modelov.

Rezultat zgornjega procesa so modeli, ki so sposobni napovedati razred TRG na poljubnih zapisih. Zapisi morajo imeti enake atribute kot učni podatki. V nadaljevanju so za posamezne tehnike zapisana pravila.

1. Odločitvena drevesa:
   Pravila, ki sestavljajo odločitveno drevo, si lahko predstavljamo hierarhično. Vozlišča predstavljajo atribute, listi pa posamezne primere, v našem primeru stranke, ki pripadajo razredu TRG.
   Zgenerirano odločitveno drevo ima 5 nivojev vozlišč (slika 18). Pomembnejši atributi so bližje korenskemu vozlišču. Najpomembnejši atribut je segment plače oz. pokojnine. Na drugem nivoju je atribut, ki pove, koliko časa je minilo od zadnje sklenitve depozita. Preostali nivoji so določeni z atributi: starost, segment sredstev in obveznosti, število sklenjenih depozitov v zadnjem letu, uporaba internetnega bančništva, uporaba osebnega bančništva in oznaka kreditnih transakcij.
   Modra barva v vozliščih označuje negativni razred (TRG = 0), rdeča pa pozitivni razred (TRG = 1). Iz slike je razvidno, da bodo odzivne stranke tiste v visokim osebnim dohodkom, velikim številom sklenjenih depozitov v preteklosti ter veliko količino sredstev.



Slika 18: Odločitveno drevo

2. Nevronske mreže

   Nevronsko mrežo sestavljajo tri vrste nevronov: vhodni, skriti in izhodni. Vsak nevron ima 1:n vhodov in 1:m izhodov. Vsak izhod je nelinearna funkcija, ki se izračuna kot vsota vhodov v nevron. Vsak vhod potuje od vhodnega nevrona preko skritih nevronov do izhodnega nevrona [17].

   Vsak vhod v skriti ali izhodni nevron ima dodeljeno utež, ki pove moč vpliva vhoda na ta nevron. Na začetku je vsakemu nevronu dodeljena določena utež. Skozi vsako iteracijo učenja modela se uteži spreminjajo. To traja toliko časa, dokler se natančnost ne spreminja več.

   Za naučeni model velja naslednje:

   – Vhodnih nevronov je 69 (za vsako diskretno stanje vhodnih atributov se vzame en vhodni nevron)
   – Skritih nevronov je 46
   – Izhodna nevrona sta dva, kolikor je izhodnih razredov

   Pomembno pri nevronski mreži s stališča platforme SQL Server 2008 je, da pri pregledovanju rezultatov ne moremo uporabiti operacij vrtanja v podatkih ali strukturnega jezika PMML (ang. predictive model markup language).

3. Logistična regresija

   Algoritem je v orodju Analysis Services izdelan kot poenostavljena nevronska mreža. Zanj veljajo enaka pravila, razlika je v tem, da nima skritih nevronov. V splošnem to pomeni, da lahko algoritem spusti določene informacije, ki bi lahko izboljšale napovedno moč modela.

4. Naivni Bayes

   Algoritem temelji na Bayesovem izreku, ki ga opisuje enačba (4.1.)

$$P(Y|X) = \frac{P(X|Y) * P(Y)}{P(X)} \quad (4.1)$$

   V primeru, da poznamo apriorno verjetnost spremenljivk X in Y in pogojno verjetnost P(X|Y), lahko izračunamo pogojno verjetnost P(Y|X).

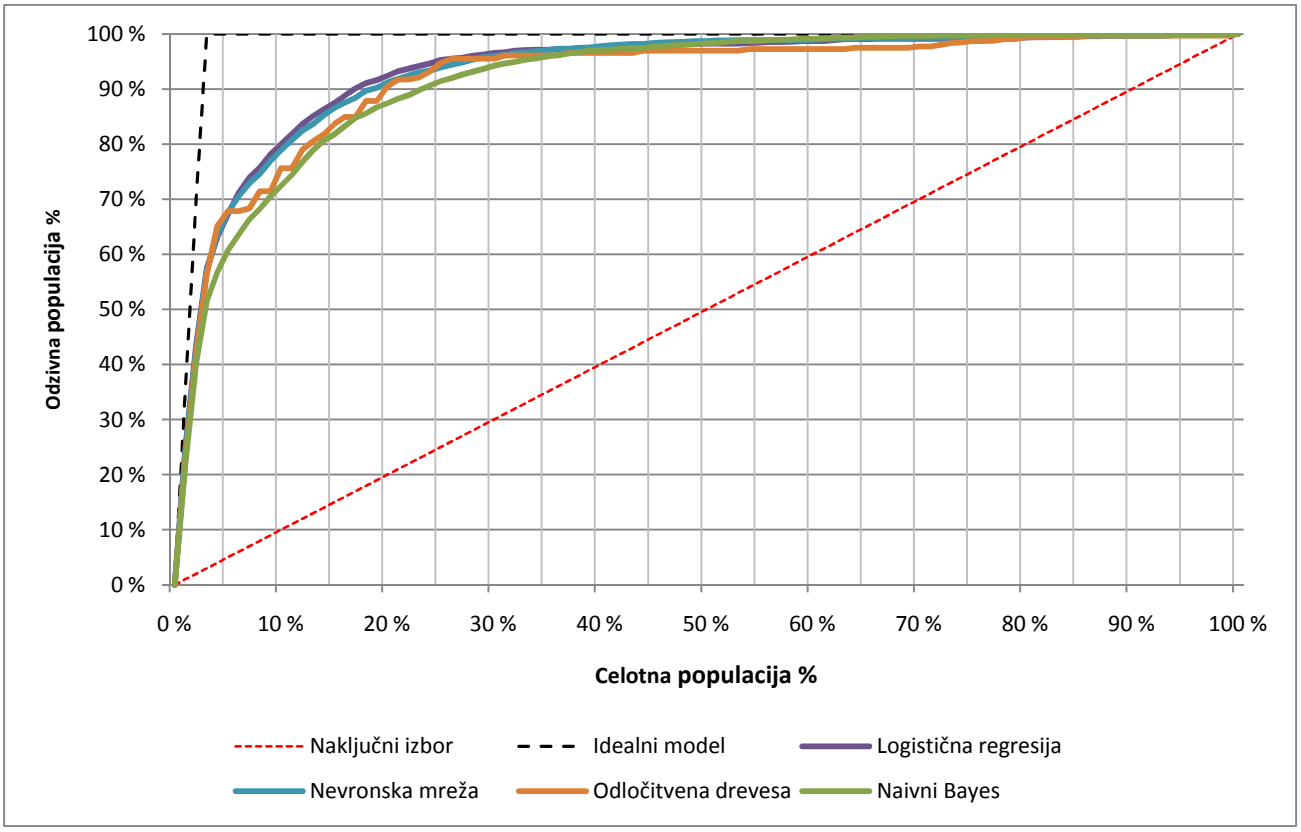
   Algoritem naivno predpostavlja, da so vsi vhodni atributi med seboj neodvisni. Izračun vrednosti razreda poteka tako, da se izračunajo pogojne verjetnosti atributov za vsak razred. Razred z najvišjo verjetnostjo je izhodni za obravnavani primer.

6.2.4.4 Ocenitev modelov

V tem koraku primerjamo modele med seboj po natančnosti. Natančnost grafično prikažemo z uporabo odzivnih diagramov, koristnostih tabel in 10-kratnega križnega preverjanja. Za preverjanje natančnosti sem uporabil drugačen nabor strank, da bi izločil možnost prilagoditve modelov učni množici podatkov.

Slika 19 prikazuje odzivni diagram vseh 4 modelov. Diagram je tabelarično predstavljen v tabeli 1, ki prikazuje skupno korist (ang. cumulative gain), faktor odziva (ang. lift) in odziv na

populaciji. Faktor odziva pove za kolikokrat se poviša verjetnost sklenitve depozita v primerjavi z naključnim izborom pri določenem odstotku populacije.



Slika 19: Odzivni diagram za izdelane modele na novih podatkih

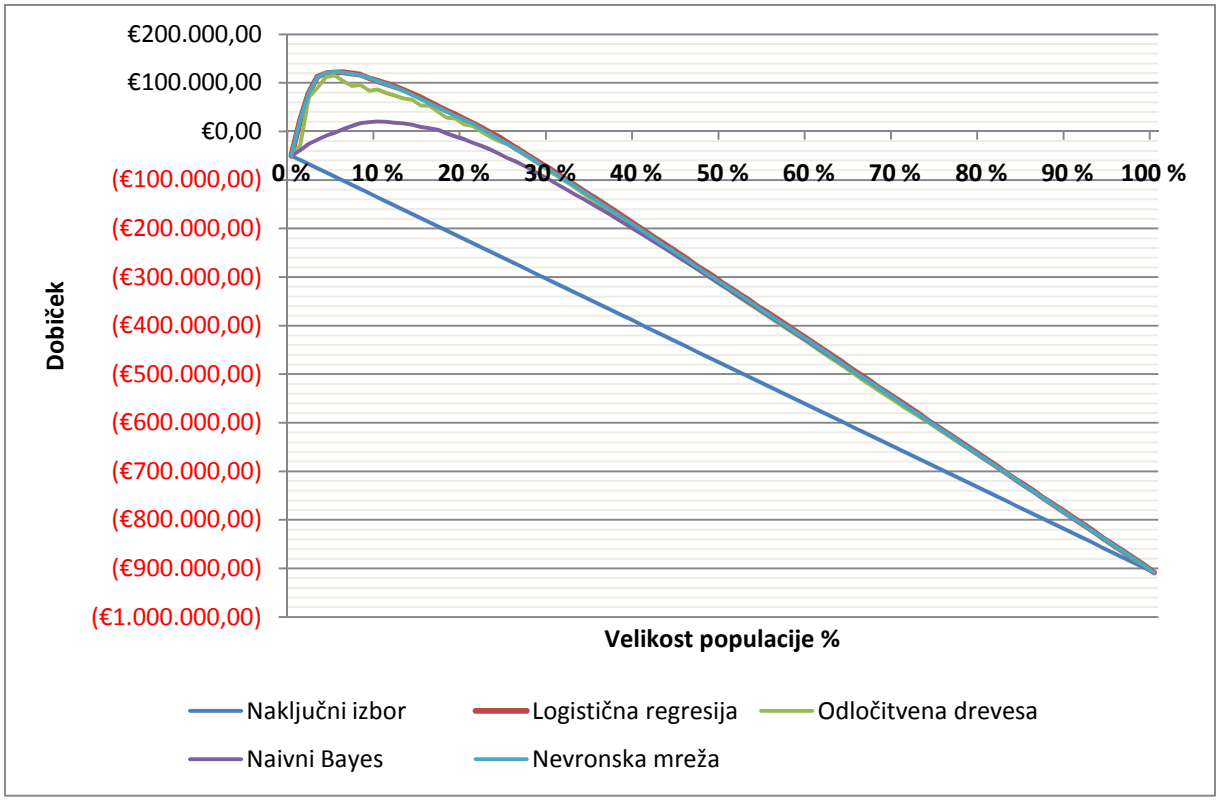
	IDEALNI MODEL	LOGISTIČNA REGRESIJA			NEVRONSKE MREŽE			ODLOČITVENA DREVESA			NAIVNI BAYES		
POPULACIJA	SKUPNA KORIST	SKUPNA KORIST	FAKTOR ODZIVA	ODZIV NA POPULACIJI	SKUPNA KORIST	FAKTOR ODZIVA	ODZIV NA POPULACIJI	SKUPNA KORIST	FATKOR ODZIVA	ODZIV NA POPULACIJI	SKUPNA KORIST	FAKTOR ODZIVA	ODZIV NA POPULACIJI
1 %	35,16 %	24,87 %	24,84	70,73%	24,17 %	24,14	68,74%	22,77%	22,74	64,76%	21,76%	21,73	61,89%
2 %	70,27 %	43,83 %	21,90	62,37%	42,91 %	21,44	61,06%	42,08%	21,03	59,88%	40,28%	20,13	57,32%
3 %	100,00 %	57,62 %	19,20	54,67%	56,92 %	18,97	54,01%	56,30%	18,76	53,43%	51,44%	17,14	48,82%
4 %	100,00 %	64,67 %	16,16	46,03%	63,53 %	15,88	45,22%	65,15%	16,28	46,37%	56,74%	14,18	40,39%
5 %	100,00 %	69,40 %	13,88	39,52%	67,60 %	13,52	38,49%	67,82%	13,56	38,62%	60,68%	12,13	34,55%
6 %	100,00 %	72,50 %	12,08	34,41%	70,49 %	11,75	33,45%	67,82%	11,30	32,18%	63,62%	10,60	30,19%
7 %	100,00 %	75,04 %	10,72	30,53%	72,90 %	10,41	29,65%	68,30%	9,76	27,78%	66,33%	9,47	26,98%
8 %	100,00 %	76,97 %	9,62	27,40%	74,65 %	9,33	26,57%	71,45%	8,93	25,43%	68,26%	8,53	24,29%
9 %	100,00 %	78,77 %	8,75	24,92%	76,97 %	8,55	24,35%	71,45%	7,94	22,61%	70,53%	7,84	22,32%
10 %	100,00 %	80,69 %	8,07	22,98%	78,90 %	7,89	22,47%	75,66%	7,56	21,54%	72,42%	7,24	20,62%
20 %	100,00 %	92,34 %	4,62	13,15%	91,20 %	4,56	12,99%	90,50%	4,52	12,89%	87,43%	4,37	12,45%
30 %	100,00 %	97,20 %	3,24	9,23%	95,97 %	3,20	9,11%	95,53%	3,18	9,07%	94,22%	3,14	8,94%
40 %	100,00 %	98,47 %	2,46	7,01%	97,77 %	2,44	6,96%	96,54%	2,41	6,87%	97,07%	2,43	6,91%
50 %	100,00 %	98,95 %	1,98	5,64%	98,73 %	1,97	5,62%	96,98%	1,94	5,52%	98,29%	1,97	5,60%
60 %	100,00 %	99,52 %	1,66	4,72%	98,95 %	1,65	4,70%	97,24%	1,62	4,62%	99,12%	1,65	4,70%
70 %	100,00 %	99,61 %	1,42	4,05%	99,26 %	1,42	4,04%	97,72%	1,40	3,98%	99,61%	1,42	4,05%
80 %	100,00 %	99,74 %	1,25	3,55%	99,43 %	1,24	3,54%	99,47%	1,24	3,54%	99,82%	1,25	3,55%
90 %	100,00 %	99,91 %	1,11	3,16%	99,61 %	1,11	3,15%	99,69%	1,11	3,15%	99,91%	1,11	3,16%

100 %	100,00 %	100,00 %	1,00	2,85%	100,00 %	1,00	2,85%	100,00%	1,00	2,85%	100,00%	1,00	2,85%

Tabela 1: Tabelarični prikaz skupne koristi in odziva (n = 80.202)

Odzivni diagram na vodoravni osi prikazuje odstotek populacije strank, na navpični osi pa skupno korist. Za model logistične regresije to pomeni, da bi z naborom 10% strank zajeli 80% takšnih strank, ki bi depozit dejansko sklenile. Pričakovano so rezultati najslabši pri modelu naivnega Bayesa, ker algoritem ne upošteva odvisnosti med vhodnimi atributi.
Logistična regresija se odreže nekoliko bolje kot nevronske mreže, kar samo pomeni, da kompleksna povezava atributov z več skritimi nivoji ni potrebna in da se nevronske mreže preveč prilagodijo učni množici. Poudariti je potrebno, da so nastavitve algoritma privzete in bi se z ustrezno nastavitvijo spremenil tudi rezultat.
Odzivni diagram lahko predstavimo v obliki diagrama dobičkonosnosti. Za izris diagrama je potrebno navesti dodatne parametre, ki opisujejo stroške in koristi od prodaje:

- Populacija: 800.000 strank
- Fiksni stroški: 50.000 €
- Strošek klica stranke: 1,5 €
- Prihodek na odzivno stranko: 15 €



Slika 20: Odzivni diagram za izdelane modele na novih podatki

Iz slike je razvidno, da je dobiček najvišji pri tistem odstotku populacije, ki je enak odstotku odzivnih strank, kar pomeni okoli 5%. Z uporabo algoritma naivni Bayes bi bi dosežen ekonomski učinek relativno slab, čeprav tega odzivni diagram ne pokaže. Zaradi majhnega števila odzivnih strank v populaciji, mora biti algoritem čim bolj natančen, sicer spremenljivi stroški hitro presežejo skupni prihodek.

Križno preverjanje se pogosto uporablja za merjenje natančnosti klasifikacijskih algoritmov. Pri križnem preverjanju originalno učno množico razdelimo v n podmnožic. Za vsako podmnožico ponovimo naslednji postopek:

1. Izbrano podmnožico uporabimo za indukcijo pravil.
2. Preostale podmnožice uporabimo za testiranje pravil.

Na koncu postopka izračunamo naslednje ocene:

- Število pravilno in nepravilno klasificiranih primerov, kjer računamo povprečje in standardni odklon. Večje število pravilno klasificiranih primerov pomeni boljši model.
- Povprečje logaritmov vseh verjetnosti napovedi, ki ga opisuje enačba (4.2). Višja vrednost pomeni, da model z večjo verjetnostjo napoveduje dogodke.

$$Log\ score = \frac{\sum_{i=1}^{n} log\ P(i)}{n} \quad (4.2)$$

- Povprečje logaritmov vseh napovedi z upoštevanjem apriorne verjetnosti napovedanega stanja, ki ga opisujejo enačbe (4.3 – 4.5). V primerjavi s prejšnjo, ta enačba nagrajuje tiste verjetnosti, ki so višje od apriorne, in močno kaznuje tiste, ki so nižje.

$$Lift = \frac{\sum_{i=1}^{n} log\left(\frac{P(i)}{Prior(Si)}\right)}{n} \quad (4.3)$$

$$Prior\ (TRG = 1) = \frac{n\ (TRG = 1)}{n} \quad (4.4)$$

$$Prior\ (TRG = 0) = \frac{n\ (TRG = 0)}{n} \quad (4.5)$$

- Kvadratni koren povprečne kvadratne napake, ki ga opisuje enačba (4.6). Nižja vrednost pomeni manjšo napako. Enačba velja za binarne klasifikatorje.

$$RMSE = \frac{\sqrt{\sum_{i=1}^{n}\left(1 - P(i)\right)^2}}{n} \quad (4.6)$$

Rezultati za 10-kratno križno preverjanje so zbrani v tabeli 2. Modeli so razvrščeni po vrstnem redu, glede na vrednosti ocen.

		Klasifikacija									
		Pravilno		Nepravilno		**Log score**		**Lift**		**RMSE**	
#	Model	povprečje	std. dev	povprečje	std. dev	povprečje	std. dev	povprečje	std. dev	povprečje	std. dev
1	**odl. dr.**	8258,8000	9,9979	193,6000	10,0716	-0,0712	0,0018	0,0635	0,0019	0,0733	0,0012
2	**nev. mr.**	8245,1997	9,9677	207,2004	10,2350	-0,0752	0,0032	0,0595	0,0032	0,0771	0,0015
3	**log. reg.**	8234,6002	13,8649	217,7999	13,7824	-0,1038	0,0050	0,0309	0,0051	0,1146	0,0053
4	**naivni B.**	7780,5000	18,2768	671,9000	18,2560	-0,1586	0,0040	-0,0239	0,0041	0,1341	0,0011

Tabela 2: Razvrstitev modelov rezultatih ocen križnega preverjanja (n = 84.510)

Primerjava zgornje tabele z odzivnim diagramom kaže, da se kvaliteta modelov ne ujema. Razlog je v tem, da odzivni diagram prikazuje rezultate na drugačni množici podatkov, kot so bili uporabljeni za izdelavo modela. To pomeni, da se algoritma nevronske mreže in odločitvena drevesa preveč prilagodita učni množici podatkov, zato bi bilo najbolje izbrati algoritem logistična regresija.

6.2.5. Vrednotenje rezultatov

Primerjava algoritmov je pokazala, da je za produkcijsko okolje najbolje uporabiti model logistične regresije. V tej fazi želimo ugotoviti, kako dobro so rezultati usklajeni s poslovnimi cilji. Na koncu določimo še nadaljnje korake.

Cilj projekta je bil izdelati model, ki bo sposoben napovedovati verjetnost sklenitve depozita bolje, kot če bi stranke izbrali naključno. Rezultati modeliranja kažejo, da smo, skozi več iteracij izbiranja atributov, prišli do modela, ki pri izboru 10% strank napoveduje 8-krat bolje kot naključni model, pri 5% strank pa 14-krat bolje. To ocenjujem kot dovolj dober rezultat za nadaljevanje v zaključno fazo – uporabo rezultatov.

6.2.5.1 Pregled procesa

Pregled procesa pokaže, ali smo slučajno izpustili pomembne informacije. V nadaljevanju kronološko navajam, kako so oddelčni sestanki, na katerih sem predstavljal opravljeno delo, vplivali na izvedbo procesa. Na sestankih so bili prisotni: vodja projekta Podatkovno skladišče, predstavnik oddelka za trženje in tehnolog. Sestanki so potekali od 18.6.2009 do 15.10.2009.

#	Tema sestanka	Vpliv na projekt v smislu korakov metodologije CRISP-DM
1	Uvodni, pred začetkom dela	• Poslovno ozadje • Ocena trenutnega stanja na področju podatkovnega rudarjenja
2	Cilji projekta	• Poslovni cilji • Grobi cilji podatkovnega rudarjenja • Kriteriji uspešnosti
3	Varnost, dostopi in arhitektura sistema	• Prvine poslovnega procesa • Projektna tveganja in možni slučaji
4	Pregled relacijskega modela	• Združevanje podatkov iz tabel podatkovne baze DB2 • Opis značilnosti podatkov
5	Pregled podatkovnih tipov	• Ocena kakovosti podatkov • Opis značilnosti podatkov
6	Pregled atributov za modeliranje	• Izbor podatkov • Izdelava podatkovne strukture
7	Pregled atributov za izdelavo končne tabele	• Integracija zapisov
8	Pregled rezultatov modela (1. iteracija)	• Načrt modeliranja (vzorčenje zapisov) • Izbor podatkov (časovna dinamika napovedovanja) • Integracija zapisov (prilagoditev atributov, agregiranje)
9	Predstavitev projekta (2. iteracija)	• Ocenitev modelov (dogovor o nadaljnjem testiranju logistične regresije) • Vrednotenje rezultatov (uspešnost modela)

		• Pregled procesa
10	Zaključni sestanek	• Določitev naslednjih korakov (izvoz rezultatov v SSIS) • Spremljanje in vzdrževanje (periodično preverjanje kvalitete modela)

Iz tabele je razvidno, da so vplivi sestankov usklajeni s fazami CRISP-DM. To ne preseneča, saj metodologija sledi naravnemu procesu izdelave projektov te vrste. Sestanki so vplivali predvsem na fazi združevanja in izbora podatkov. Spremembe, ki sem jih izvajal v sklopu teh dveh faz, so tudi dejansko imele največji vpliv na točnost napovedovanja modelov.

6.2.5.2 Določitev naslednjih korakov

V tem koraku se odločamo, ali je mogoče model uporabiti na produkcijskih podatkih. Zaradi pogostega ponavljanja celotnega procesa, sem do tega koraka prišel šele, ko je bil model dovolj dober in sprejemljiv za uporabo v konkretni aplikaciji.

6.2.6. Uporaba rezultatov

6.2.6.1 Načrtovanje razvoja

Uporaba rezultatov modela je odvisna od organizacije, ki bo model uporabljala. V primeru banke, bosta rezultate uporabljala:

- Informacijski sistem za delo na bančnem okencu in
- klicni center na oddelku za trženje

V prvem primeru bodo programerji rezultate preko vgrajenih procedur (ang. stored procedure) iz podatkovne baze dobili podatke za prikaz na zaslonskih slikah, v drugem pa seznam podatkov (ime, priimek, telefon) npr. 1.000 najbolj potencialnih strank.

V domeni projekta moramo model vsak mesec zagnati na parametrizirani tabeli strank. Zaradi uporabe podatkovnega skladišča se nekateri podatki (presečni podatki o stranki) preračunajo in zapišejo 1. v mesecu, nekateri (plača, sredstva, ipd) pa 15. v mesecu. To pomeni, da lahko zaženemo izdelavo parametrizirane tabele šele 16. v mesecu, pri tem pa računamo verjetnost sklenitve depozita do prihodnjega meseca. Primer: 16.10.2009 poženemo model, ki bo izračunal verjetnost sklenitve depozita za obdobje 16.10.2009 do 15.11.2009. To pravilo mora biti za končnega uporabnika nevidno.
Uporabnika rezultatov pričakujeta naslednje vhodne podatke:

- Identifikator stranke
- Identifikator produkta. V tem primeru je to enolična številka, saj napovedujemo samo eno vrsto produkta, to je dolgoročni depozit.
- Verjetnost sklenitve depozita. Vrednost je interval [0, 1], pri čemer vrednost 0 pomeni, da stranka zagotovo ne bo sklenila depozita, in 1 da ga bo.

Za zgornje podatke bom uporabil programsko orodje SSIS, ki bo izvedel celoten proces ETL: pridobivanje podatkov za parametrizirano tabelo; zagon modela in izračun verjetnosti; zapis v

tabelo naklonjenosti stranke k sklenitvi depozita (ang. propensity scoring). Tabela bo poleg zgoraj naštetih atributov vsebovala še:

- Datum zadnjega zagona modela in
- zaporedno številko iteracije modela, saj pričakujemo, da se bo model v prihodnosti spreminjal.

Zgornja dva podatka omogočata spremljanje in vzdrževanje modela.

6.2.6.2 Strategija spremljanja in vzdrževanja

Po koncu projekta se modeli uporabljajo s periodičnim zagonom zgoraj omenjenega procesa ETL. Za učinkovito vzdrževanje to pomeni, da moramo pri vsakem zagonu projekta SSIS vedeti, kaj se je med izvajanjem dogajalo:

- Koliko časa je trajal celoten prenos?
- Ali je prišlo do napak pri poizvedbah?
- Ali je kakšen del procesa trajal predolgo in zakaj?

Z zagotovitvijo teh informacij je vzdrževanje ob napakah relativno preprosto. Praksa je pokazala, da je projekt SSIS najbolje organizirati tako, da vsaka podatkovna tabela predstavlja objekt, transformacija (brisanje, posodabljanje, branje) na njej pa metodo. Pred in po vsakem zagonu metode, zapišemo v dnevnik (ang. log) datum in čas začetka izvajanja operacije. Za vsak nepričakovan dogodek (ang. event), prav tako sprožimo pisanje v dnevnik. Bančni sistem že uporablja preprost pregledovalnik dnevnika, zato je potrebno le še zapisati datum in čas v pravilne tabele.

Spremljanje modela je bolj semantične narave. Zanima nas, ali bo model npr. čez 1 leto še vedno uspešno napovedoval sklenitve depozitov. Za oba uporabnika modelov je bilo že v samem začetku projekta predviden zapis odziva v odzivno tabelo (ang. feedback area). Odzivna tabela vsebuje naslednje podatke:

- Identifikator stranke
- Datum odziva
- Odziv stranke, ki je lahko pozitiven, negativen ali pogojno pozitiven (npr.: »Depozit bom sklenil čez 3 mesece«)

Te podatke lahko npr. vsake dva meseca uporabimo za ponovno izdelavo modela, ki upošteva tudi odziv stranke. S temi podatki lahko naredimo tudi obratno napoved, npr. strank, ki depozita zagotovo ne bodo sklenile in to upoštevamo pri zagonu modela.

6.2.6.3 Končno poročilo

Poročilo za končnega uporabnika sem izdelal v obliki predstavitve Powerpoint, ki je naveden v prilogi.

6.2.6.4 Revizija projekta

Pri ponovnem pregledu projekta sem ugotovil, da uporaba metodologije CRISP-DM vodi do uspešnega zaključka projekta. V tem primeru do rezultatov, ki so bistveno natančnejši, kot pa če bi izbrali naključen nabor strank. Izkušnje na projektu so pokazale, da so ključi za uspeh naslednji:

1. Poznavanje poslovnih procesov:
   - Kako funkcionira oddelek za trženje?
   - Kdaj so stranke najbolj dovzetne za sklepanje storitev?
2. Poznavanje podatkovnih skladišč:
   - Kdaj in kako se polnijo tabele?
   - Kaj vse se upošteva pri izračunavanju atributov (npr. plače, sredstev, ipd)?
3. Poznavanje pomena podatkov. S tem se znebimo nepomembnih atributov, čeprav se sprva zdi, da so pomembni in dobri vsi. Uporaba lokalnih segmentov, ki veljajo v Sloveniji, je nadalje izboljšala natančnost napovedovanja modela.
4. Vključitev končnih uporabnikov v proces. V tem primeru to ni bilo težko, saj je npr. za klicni center bistvenega pomena, ali se bo stranka pozitivno odzvala na klic ali ne.

Posebnih poročil med nastajanjem projekta nisem zapisoval, saj sem potek dela sproti opisoval na sestankih. Vpliv teh sestankov sem zapisal poglavju »Pregled procesa«.

7. Podatkovno rudarjenje z orodjem Weka

7.1. Uvod

Razvoj novozelandskega programa za podatkovno rudarjenje WEKA (Waikato Environment for Knowledge Analysis) se je pričel leta 1992 na univerzi Waikato. Z vključitvijo novozelandske vlade leta 1993 je postal cilji projekta izdelati vrhunsko platformo za hitro razvijajoče tehnike iz podatkovnega rudarjenja in raziskati njegovo uporabo [18]. Trenutna verzija programa je 3.6.1., koda pa je v celoti spisana v Javi. Platformo sestavlja več orodij, s katerimi lahko beremo in analiziramo podatke ter izvajamo podatkovno rudarjenje.

Glede na okvir diplomske naloge, v kateri poudarjam poslovni vidik podatkovnega rudarjenja, je izrednega pomena povezljivost med različnimi viri podatkov. Prav tako nas zanima hitrost razvoja in povezljivost z analitičnimi orodji. Za poslovnega analitika, ki se ne ukvarja s tehničnimi podrobnostmi programske opreme, je izredno pomembno, da lahko čim prej pride do uporabnih rezultatov.

7.2. Izvedba postopka

V nadaljevanju so opisane samo posamezne faze, ki so ključne za praktično primerjavo obeh orodij.

7.2.1. Analiza in priprava podatkov

Po metodologiji CRISP-DM sta analiza in priprava podatkov tisti dve fazi, ki vzameta največ časa za izvedbo. Pri obeh se delo prične z branjem podatkov iz poljubne podatkovne zbirke. Weka v ta namen uporablja vmesnik JDBC (ang. Java database connectivity). Na tak način lahko načeloma dostopamo do poljubne podatkovne baze, ki ima za vmesnik JDBC napisan gonilnik. V našem primeru so podatki shranjeni v podatkovni bazi SQL Server 2008, za katero omenjen gonilnik obstaja.

Namestitev gonilnika in vnos povezovalnega niza nista bila dovolj, saj povezava kljub temu ni delovala. Razlog je v tem, da ima Weka v eni od konfiguracijskih datotek v arhivu JAR zapisane podatkovne tipe za posamezne podatkovne baze. Ker so bili podatkovni tipi pomanjkljivo zapisani, je bilo potrebno ročno popraviti konfiguracijsko datoteko in ponovno izdelati arhiv JAR. Ker bi takšen poseg v programsko kodo nestrokovnjaku onemogočal pričetek dela, sem poskusil učno tabelo iz podatkovne baze SQL Server prenesti v tekstovno datoteko CSV (ang. comma separated values). Ta datoteka velikosti 13 MB je vsebovala približno 80.000 vrstic. Weka datoteke ni uspela prebrati v celoti. Program je, po nekaj minutnem branju, vrnil napako *OutOfMemoryException*, zaradi presežene porabe pomnilnika, ki je bila nastavljen na 1GB.

7.2.2. Modeliranje

Za primerjavo s platformo SQL Server 2008 (Analysis Services) sem uporabil orodje KnowledgeFlow ali načrtovalec znanja. Gre za grafično orodje, s katerim sestavimo tok podatkov, in omogoča:

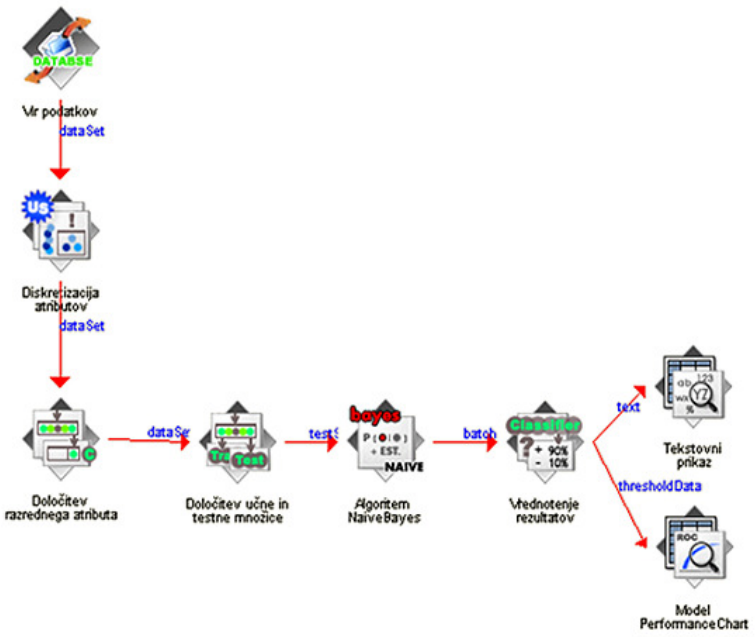
- izbiro vhodnega vira npr. iz podatkovne baze
- frekvenčno analiza podatkov
- delitev podatkov na učno in testno množico
- izbiro algoritmov za klasifikacijo, razvrščanje v skupine in izdelavo asociacijskih pravil
- grafični ali tekstovni prikaz rezultatov

Proces sestavimo iz komponent. Izhod vsake komponente lahko povežemo na drugo komponento, če ta že nima povezave oz. če je njen vhod kompatibilen. Komponente se delijo v 8 kategorij:

1. Vir podatkov, ki jih podpira Weka: podatkovna baza, tekstovna datoteka CSV, ipd.
2. Ponor podatkov, kamor lahko zapišemo rezultat.
3. Predprocesiranje: diskretizacija, dodajanje in odstranjevanje atributov, prevzorčenje, ipd.
4. Klasifikatorji: vsi algoritmi za klasifikacijo.
5. Algoritmi za razvrščanje v skupine.
6. Algoritmi za računanje asociacijskih pravil.
7. Delitev podatkov na učno in testno množico in merjenje kakovosti modelov.
8. Prikaz rezultatov: tekstovno, grafično.

V Weki je implementirano veliko število algoritmov. Samo za klasfikacijo denimo več kot 30. To pomeni, da uporaba ni omejena na zgolj reševanje poslovnih problemov, temveč na vsa strokovna področja.

V Weki sem ponovil izdelavo modelov z algoritmi, ki sem jih uporabljal v orodju Analysis Services. Sama izdelava procesa iz komponent je preprosta in intuitivna. Primer procesa za izdelavo modela prikazuje slika 20. Uporabil sem algoritem naivni Bayes. Rezultate lahko interpretiramo z grafično ali tekstovno komponento.



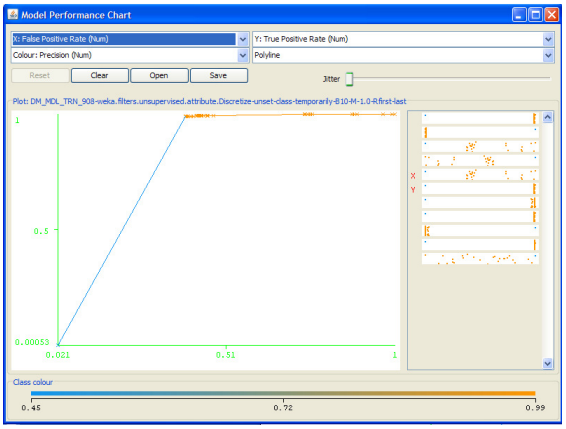
Slika 21: proces podatkovnega rudarjenja v orodju KnowledgeFlow

V nadaljevanju so opisani rezultati posameznih modelov.

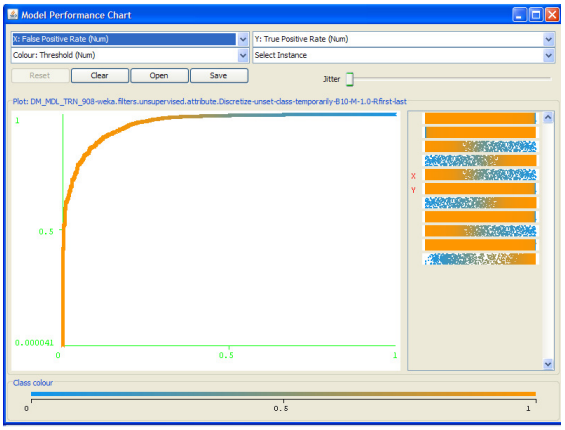
7.3. Primerjava rezultatov

V Weki sem po opisanem procesu izdelal modele z uporabo algoritmov, ki po funkcionalnosti ustrezajo tistim iz SQL Server 2008. V nadaljevanju za vsakega posebej prikazujem diagram učinka in tabelo statističnih ocen. Za modeliranje sem uporabil enako množico podatkov kot prej. Učna in testna množica sta v razmerju 70:30.
Odzivni diagrami za modele so naslednji:

Odločitvena drevesa:

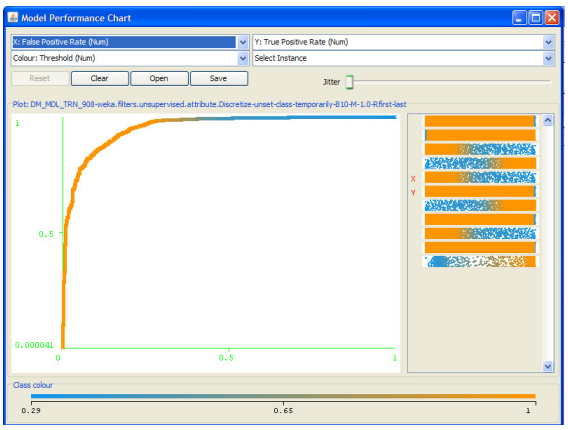


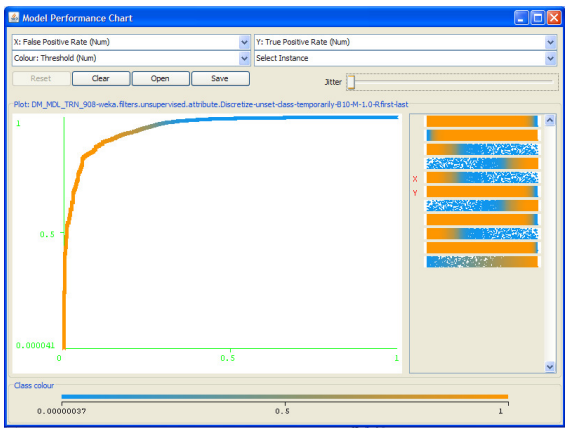
Logistična regresija:



Nevronske mreže:

Naivni Bayes:





Statistične ocene opisuje naslednja tabela.

	Odločitvena drevesa	Logistična regresija	Nevronske mreže	Naivni Bayes
Natančnost	97,8862%	97,902%	97,9256%	95,2636%
Točnost	0,689	0,703	0,703	0,361
Občutljivost	0,545	0,527	0,541	0,739
$F_{\beta=1}$	0,609	0,602	0,612	0,485
RMSE	0,134	0,1289	0,1302	0,1975

7.4. Ugotovitve

Iz tabele statističnih vrednosti lahko sklepamo, da so algoritmi v Weki za okoli 5-10% boljši v točnosti in občutljivosti od tistih v orodju Analysis Services. Pri tem je potrebno upoštevati, da je učenje modelov v Weki za nekaj stopenj počasnejše. Za učenje modela z nevronskimi mrežami je potrebno ustrezno nastaviti parametra *learningRate* in *trainingTime*, da algoritem sploh dokonča proces v sprejemljivem času. Število strank za učenje modela v tem primeru ni presegalo 100.000. Pri podatkovnem rudarjenju v velikih družbah lahko ta številka hitro preseže nekaj 10 milijonov. Če bi želeli npr. izvesti kompleksnejšo analizo na vseh strankah, bi bilo izvajanje bistveno počasnejše. Algoritme v Weki bi lahko parametrizirali za hitrejše izvajanje, kar pa bi imelo negativno posledico v slabši natančnosti.
V Analysis Services težav s počasnim učenjem modelov ni bilo, kar daje vtis robustnosti tudi pri veliki količini podatkov. Potrebno je upoštevati, da sem kot vir podatkov uporabil SQL Server, kar pomeni, da je dostop do podatkov optimiziran.
Število algoritmov, ki jih podpira orodje Analysis Services, je premajhno za reševanje vseh problemov, ki se s podatkovnim rudarjenjem rešujejo. Uporabljamo ga lahko predvsem za tipične poslovne probleme. SQL Server ima veliko prednost tudi v samem prenosu podatkov. Z uporabo orodja SSIS lahko ne samo prenesemo podatke temveč jih tudi poljubno preoblikujemo. Zapis rezultatov nazaj v podatkovno bazo prav tako izvedemo z eno od komponent v SSIS.

Za uporabnika, ki se bo podatkovnega rudarjenja lotil brez dobrega strokovnega predznanja, je pomembna tudi krivulja učenja. V Weki je ta pozitivno pospešena, saj šele po določenem času postane orodje intuitivno za uporabo. Po tem času program ponuja ogromno možnosti za uporabo različnih algoritmov in vizualizacijo rezultatov. Analysis Services po drugi strani je sprva preprost za uporabo, a je kasneje omejen pri funkcionalnostih. Oba programa bi lahko uporabljali tudi vzajemno. Z uporabo orodja SSIS podatke naložimo in pretvorimo v pravilno obliko. V Weki nato uporabimo primeren algoritem za izdelavo modela in podatke zapišemo nazaj v podatkov bazo.

SQL Server omogoča integracijo modelov v Microsoftova orodja, kot sta Excel in Visio. V Excelu lahko celo izdelujemo modele, čistimo podatke in izvajamo frekvenčno analizo atributov. Rezultate lahko shranimo na strežnik Sharepoint. Do njih lahko dostopamo s programi, ki se znajo povezati na tak strežnik; najbolj pogost primer je pregled podatkov z Excelom.

Strukturo modelov lahko izvozimo v obliki strukture PMML, ki jo lahko uvozimo v orodjih drugih proizvajalcev, kot so IBM DB2, SAS Enterprise Miner in SPSS. Weka ne podpira jezika PMML, lahko pa preko vtičnikov (ang. plugin) dostopamo do njenih algoritmov. Te vtičnike podpirajo odprtokodna orodja KNIME, RapidMiner in R. Integracijo Weke v poslovno uporabo omogoča program Pentaho BI Suite [18].

Iz opisanega sledi, da je osnovna izvedba Weke bolj primerna za akademsko raziskovanje, za konkretne poslovne probleme pa Analysis Services.

8. Zaključek

Podatkovno rudarjenje se je v zadnjem desetletju vzpostavilo kot povsem samostojno področje raziskovanja. Zlasti v gospodarstvu lahko njegova uporaba pripomore k zmanjšanju stroškov, kar za družbo posledično pomeni večji dobiček. Na projektu podatkovnega rudarjenja lahko pridemo do prihrankov na tri načine:

1. Skrajšamo čas za izvedbo projekta.
2. Uporabimo manjšo skupino zaposlencev.
3. Prenesemo delo na znanjske delavce z nižjo stopnjo izobrazbe.

K prvima dvema pristopoma težijo vsi proizvajalci orodij za podatkovno rudarjenje. Microsoft se s platformo SQL Server nagiba tudi k tretjemu pristopu. Orodje Analysis Services je preprosto za uporabo, zato lahko hitro pridemo do rezultatov. Zaradi dobre vizualizacije uporabnik ne potrebuje dobrega predznanja iz statistike, da bi ugotovil, ali je model dober ali ne. Za lažjo uporabo je na voljo le majhno število algoritmov. S to omejitvijo je Microsoftovo orodje v osnovi uporabno predvsem za preprostejše poslovne probleme.

Med raziskovanjem so se potrdile navedbe metodologije CRISP-DM, da največ časa porabimo za pridobivanje in transformiranje podatkov v obliko, ki je primerna za izdelavo modelov. Velike družbe podatke praviloma shranjujejo v podatkovnih skladiščih različnih proizvajalcev. To lahko povzroča težave že pri samem prenosu podatkov. Weka je npr. imela zaradi količine podatkov težave pri uvozu preprostih tekstovnih datotek, za dostop do baze SQL pa je bilo potrebno popravljati celo kodo programa. V tem oziru so najboljša orodja tista, ki imajo vgrajeno dobro podporo za operacije ETL. Med te spada tudi SQL Server z orodjem SSIS. Za sam prenos je potrebno zelo dobro poznati strukturo osnovnih tabel. Še bolje je, če imamo na voljo kocke OLAP, v katerih so podatki že primerno združeni. Znanje o podatkovni arhitekturi imajo predvsem strokovnjaki na področju podatkovnih baz. Za razumevanje poslovnih ciljev na drugi strani potrebujemo dobre ekonomiste. Podatkovno rudarjenje v gospodarstvu je zato skupinski projekt, ki vključuje zaposlence iz različnih panog.

Med postopkom podatkovnega rudarjenja sem ugotovil, da na rezultate bistveno vpliva frekvenčna porazdelitev strank glede na vrednost klasifikatorja. S prilagajanjem vzorca se je močno spreminjala mera občutljivosti modelov. Z empiričnim testiranjem sem prišel do ugotovitve, da je za napoved depozitov najbolje uporabiti logistično regresijo. Ostali algoritmi so se ali preveč prilagodili vzorcu (odločitvena drevesa, nevronske mreže) ali pa so imeli premajhno natančnost (naivni Bayes). Omenjene algoritme je možno skonfigurirati tako, da se bolj ali manj prilagodijo podatkom. Na tem področju se je zelo izkazala Weka, v kateri lahko za vsak algoritem (teh je preko 50) natančno parametriziramo. Pri tem je seveda obvezno tehnično poznavanje algoritmov. Za oba programa velja, da ju je možno nadgraditi z novimi algoritmi.

Podatkovno rudarjenje, ki sem ga izvajal kot del projektne skupine, je zanimivo tudi z vidika samega razvoja. Treba je namreč upoštevati, da je šlo za prvi tovrstni projekt v banki. Na začetku je kazalo, da se projekt razvija iterativno oz. prototipno (tak način spodbuja tudi metodologija CRISP-DM). Zaradi nezadostnega poznavanja tehnologije in algoritmov, so bili modeli iz prvih iteracij nenatančni. Iterativni razvoj s prototipiranjem zahteva dobro znanje, saj le tako lahko poteka hitro. V primeru tega projekta pa so se zahteve razvijale sproti, tako kot tudi same rešitve. S tega vidika je šlo za agilni razvoj, ker se je delo neprestano prilagajalo razmeram (znanju) v skupini.

V banki je bil odziv na rezultate projekta pozitiven. Na produkcijskih podatkih model logistične regresije pravilno napove sklenitev depozita za približno 60% strank. Naklonjenost stranke za sklenitev depozita vpisujemo v podatkovno skladišče DB2. Do teh podatkov lahko uporabniki v banki dostopajo z namenskimi programi preko vgrajenih procedur. V bodoče tudi npr. z Excelom preko strežnika Sharepoint.

Model, ki ga opisujem v diplomski nalogi, je mogoče še izboljšati. Ena od možnih izboljšav bi bila, da bi za vsakega od segmentov strank (mladi, seniorji, ipd) izdelali svoj model in poiskali specifične lastnosti vsake starostne skupine. Druga možnost je, da bi za stranke, ki po izteku depozita ne sklenejo novega, poiskali skupne značilnosti oziroma vrednosti atributov. S temi ugotovitvami bi znali za potencialno odhajajoče stranke izdelati prilagojeno ponudbo.

9. Priloge

Diplomi prilagam predstavitev, s katero sem sodelavcem na projektu podatkovno skladišče predstavil proces podatkovnega rudarjenja in njegov pomen za banko.

10. Literatura

[1] Kononenko, I. in Kukar, M. *Machine learning and data mining.* Chichester : Horwood Publishing, Ltd., 2007.

[2] Hand, D. J., Manilla, H. in Smyth, P. *Principles of data mining.* Cambridge (Massachusetts), London : MIT Press, 2001. str. 1-39.

[3] Tan, P. N., Steinbach, M. in Kumar, V. *Introduction to data mining.* Boston : Pearson Addison Wesley, 2006.

[4] MacLennan, J., Tang, Z. H. in Crivat, B. *Data Mining with Microsoft SQL Server 2008.* Indianapolis : Wiley Publishing, Inc., 2009.

[5] Giudici, P. in Figini, S. *Applied Data Mining for Business and Industry.* 2nd ed. Wiley, 2009. str. 1-39.

[6] Inmon, W. H. *Building the Data Warehouse.* 3rd Edition. New York : John Wiley & Sons, Inc., 2002. str. 1-29.

[7] Codd, E.F., Codd, S. B. in Salley, C.T. [Elektronski] http://www.minet.uni-jena.de/dbis/lehre/ss2005/sem_dwh/lit/Cod93.pdf.

[8] Rant, Ž. Prenos znanja kot dilema znanjskih delavcev in učeče se organizacije. *Organizacija.* 2008, Zv. 41, 2.

[9] KDNuggets. KDNuggets. [Elektronski] http://www.kdnuggets.com/polls/2007/data_mining_methodology.htm.

[10] Krisper, M., in drugi. *Enotna metodologija razvoja informacijskih, Uvod.* Ljubljana : CVI, 2003.

[11] Shearer, C. The CRISP-DM Model: The New Blueprint for Data Mining. *Journal of Data Warehousing.* 2000, Zv. V, 4, str. 13-21.

[12] Chapman, P., in drugi. *CRISP-DM.* [Elektronski] http://www.crisp-dm.org/.

[13] Ogorevc, T. *Podatkovno skladišče (BDW).* [Interni dokument (Powerpoint)] Ljubljana : NLB, d.d.

[14] Delo, d.d. [Elektronski] http://oglasi.delo.si/download/cenik2009_nov.pdf.

[15] Chawla, N. V., Japkowitz, N. in Kolcz, A. Editorial: Special Issue on Learning from Imbalanced Data. *Special Interest Group on Knowledge Discovery and Data Mining.* [Elektronski] 2004. http://www.sigkdd.org/explorations/issues/6-1-2004-06/edit_intro.pdf.

[16] Vuk, M. in Curk, T. ROC Curve, Lift Chart and Calibration Plot. *Metodološki zvezki.* 2006, Zv. 3, 1.

[17] Microsoft. Microsoft Neural Network Algorithm Technical Reference. *MSDN.* [Elektronski] http://msdn.microsoft.com/en-us/library/cc645901.aspx.

[18] Hall, M. The WEKA Data Mining Software: An Update. *Special Interest Group on Knowledge Discovery and Data Mining.* [Elektronski] 2009. http://sigkdd.org/explorations/issues/11-1-2009-07/p2V11n1.pdf.

[19] Microsoft. Cross-Validation (Analysis Services - Data Mining). *MSDN.* [Elektronski] http://msdn.microsoft.com/en-us/library/bb895174.aspx.

[20] Aligning the Warehouse and the Web. [ured.] J. Wang. *Encyclopedia of data warehousing and mining.* 2nd ed. New York : Information Science Reference, 2009, str. 18-23.
[21] Microsoft. Data Mining Algorithms (Analysis Services - Data Mining). *MSDN.* [Elektronski] http://technet.microsoft.com/en-us/library/ms175595.aspx.
[22] SAS Institute Inc. *Data Mining and the Case for Sampling.* SAS Institute Inc., 1998.
[23] Rud P., O. *Data Mining Cookbook.* John Wiley & Sons, Inc., 2001. str. 3-23.