UNIVERZA NA PRIMORSKEM
FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN
INFORMACIJSKE TEHNOLOGIJE

ZAKLJUČNA NALOGA

UPORABA PODATKOVNEGA RUDARJENJA ZA DETEKCIJO MAČK IN PSOV V SLIKAH

NIKI HROVATIN

2018
ZAKLJUČNA NALOGA
HROVATIN

UNIVERZA NA PRIMORSKEM
FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN
INFORMACIJSKE TEHNOLOGIJE

Zaključna naloga

Uporaba podatkovnega rudarjenja za detekcijo mačk in psov v slikah

(The use of Data Mining Methods to Detect Cats and Dogs in Images)

Ime in priimek: Niki Hrovatin
Študijski program: Računalništvo in informatika
Mentor: doc. dr. Branko Kavšek

Koper, avgust 2018

Ključna dokumentacijska informacija

Ime in PRIIMEK: Niki HROVATIN

Naslov zaključne naloge: Uporaba podatkovnega rudarjenja za detekcijo mačk in psov v slikah

Kraj: Koper
Leto: 2018
Število listov: 61 Število slik: 24 Število tabel: 10
Število referenc: 56

Mentor: doc. dr. Branko Kavšek

Ključne besede: Asirra, podatkovno rudarjenje, Weka, ImageFilter, prepoznavanje slik, strojno učenje, Kaggle

Izvleček:

V zaključni nalogi je predstavljena aplikacija procesa podatkovnega rudarjenja za detekcijo mačk in psov v slikah. Uporabljena podatkovna zbirka je del večje podatkovne zbirke Asirra, ki vsebuje 3 milijone digitalnih slik mačk in psov. Cilj zaključne naloge je izdelava napovedovalnega modela, ki je zmožen učinkovito razlikovati med slikami, ki prikazujejo mačke in slikami, ki prikazujejo pse. Proces podatkovnega rudarjenja se izvaja po CRISP-DM metodologiji. Za analizo podatkov, obdelavo podatkov, gradnjo modelov in testiranje modelov je uporabljen odprtokodni program Weka z vtičnikom ImageFilter, ki razširja program Weka z filtri za zajeti lastnosti iz digitalnih slik. Za pretvoriti digitalne slike v množice podatkov so bili uporabljeni naslednji filtri: PHOG filter, Edge Histogram Filter in JPEG Coefficient Filter. Ker aplikacija filtrov je proizvedla številno množico atributov smo uporabili metodo vodilnih komponent za zmanjšati število atributov in posledično skrajšati čas učenja modelov. Za izgradnjo modelov so uporabljeni naslednji klasifikacijski algoritmi: sekvenčna minimalna optimizacija, logistična regresija, naključni gozd, J48 in naivni Bayes. Modeli so testirani z uporabo N-kratnega prečnega preverjanja. Najboljše rezultate je dosegel algoritem sekvenčne minimalne optimizacije, ki je zmožen klasificirati digitalne slike mačk in psov z 80% natančnostjo.

Key words documentation

Name and SURNAME: Niki HROVATIN

Title of the final project paper: The use of Data Mining Methods to Detect Cats and Dogs in images

Place: Koper
Year: 2018
Number of pages: 61 Number of figures: 24 Number of tables: 10
Number of references: 56

Mentor: Assist. Prof. Branko Kavšek, PhD

Keywords: Asirra, data mining, Weka, ImageFilter, image recognition, machine learning, Kaggle

Abstract:

The final project paper presents the application of the data mining process to detect cats and dogs in digital images. The used data is part of a larger database named Asirra, which contains over 3 million digital images of cats and dogs. The aim of the final project is to produce a prediction model that is capable of effectively distinguishing between images that show cats and images showing dogs. The data mining process is performed according to the CRISP-DM methodology. To perform the whole process, we use the open source program Weka with the ImageFilter plugin. The ImageFilter plugin extends the Weka program with filters used to extract features from digital images. To extract features from digital images were used PHOG filter, Edge Histogram Filter and JPEG Coefficient Filter. To reduce the number of attributes generated by the application of filters we used the principal component analysis procedure. The following classification algorithms were used to build models: sequential minimal optimization, logistic regression, random forest, J48 and naive Bayes. The models were tested using N-fold cross-validation. The best results were achieved by the sequential minimal optimization algorithm that can classify digital images of cats and dogs with 80% accuracy.

ZAHVALA

Zahvaljujem se mentorju doc. dr. Branku Kavšku za svetovanje in pomoč pri izdelavi zaključne naloge.
Posebna zahvala gre moji družini in prijateljem za pomoč in spodbudo med tekom študija.

KAZALO VSEBINE

KAZALO PREGLEDNIC

KAZALO SLIK IN GRAFIKONOV

SEZNAM KRATIC

CRISP-DM	med-industrijski standardni proces za podatkovno rudarjenje (angl. Cross-Industry Standard Process for Data Mining)
CAPTCHA	popolnoma avtomatiziran Turingov test s pomočjo katerega ločimo računalnike in ljudi med seboj (angl. Completely Automated Public Turing-test-to-tell Computers and Humans Apart)
ARFF	atribut-relacija format datoteke (angl. Attribute-Relation File Format)
KDD	konferenca o odkrivanju znanja in podatkovnem rudarjenju (angl. Knowledge Discovery in Databases and Data Mining Conference)
GNU-GPL	splošno dovoljena licenca (angl. General Public License)
ASCII	ameriški standardni nabor za izmenjavo informacij (angl. American Standard Code for Information Interchange)
PHOG	piramidni histogram orientiranih gradientov (angl. Pyramid Histogram of Oriented Gradients)
HOG	histogram orientiranih gradientov (angl. Histogram of Oriented Gradients)
EHD	histogram robov (angl. Edge Histogram Filter)
DCT	diskretna kosinusna transformacija (angl. Discrete Cosine Transform)
PCA	metoda vodilnih komponent (angl. Principal Component Analisys)
SMO	sekvenčna minimalna optimizacija (angl. Sequential Minimal Optimisation)
SVM	metoda podpornih vektorjev (angl. Support Vector Machines)
QP	kvadratično programiranje (angl. Quadratic Programming)

1 UVOD

Vsako leto višja rast proizvedene količine podatkov zagotavlja znanstvenikom, inženirjem in poslovnim ljudem obsežen in bogat vir, ki je mogoče analizirati za odkrivanje finančno dragocenih vzorcev, optimizacijo industrijskih sistemov in izvršiti znanstvena odkrivanja. Podatkovno rudarjenje je proces odkritja zanimivih, nepričakovanih ali dragocenih strukturah v velikih podatkovnih zbirkah [1].

V zaključni nalogi bo predstavljen primer uporabe podatkovnega rudarjenja za detekcijo mačk in psov v digitalnih slikah. Podatki uporabljeni za izgradnjo modelov so del večje podatkovne zbirke imenovane Asirra [2] uporabljene za CAPTCHA (angl. **C**ompletely **A**utomated **P**ublic **T**uring-test-to-tell **C**omputers and **H**umans **A**part). CAPTCHA je popolnoma avtomatiziran Turingov test s pomočjo katerega ločimo računalnike in ljudi med seboj. Podatkovna zbirka Asirra vsebuje veliko zbirko digitalnih slik mačk in psov, uporabljenih za izvajanje Turingovega testa [3] nad uporabniki spletnih storitev, za preverjati če so dejansko ljudje, ki dostopajo do storitve. Prepoznavanje slik je zelo hitra in enostavna operacija za ljudi ampak je zahtevna naloga za računalnike.

V zaključni nalogi bodo uporabljene napredne metode podatkovnega rudarjenja za zgraditi čim bolj učinkovit klasifikator, ki bo skušal premagati izziv postavljen od podatkovne zbirke Asirra. Proces podatkovnega rudarjenja je izveden po CRISP-DM metodologiji [4] (angl. **C**ross-**I**ndustry **S**tandard **P**rocess for **D**ata **M**ining), ki natančno določa potek analize podatkov ter standardizira vse korake tako proces podatkovnega rudarjenja postane ponovljiv in zanesljiv. Velik del procesa podatkovnega rudarjenja se bo izvajal na programu Weka [5], kjer podatki bodo analizirani, obdelani in se bo gradilo napovedovalne modele nad njimi. Analiza podatkov se začne z pretvorbo surovih podatkov v ARFF [6] datoteko, ki je potem naložena v programu Weka. Za uporabiti metode podatkovnega rudarjenja nad podatkovno zbirko sestavljeno iz digitalnih slik je potrebno zajeti lastnosti digitalnih slik. Ta postopek bo izveden z uporabo vtičnika ImageFilter [7] s katerim bomo pretvorili digitalno sliko v množico atributov. Aplicirali bomo tri različne filtre nad podatki: PHOG filter, Edge Histogram Filter in JPEG Coefficient Filter. Po aplikaciji filtrov bomo uporabili metodo vodilnih komponent za zmanjšati število atributov generiranih z aplikacijo filtrov in posledično olajšati postopek gradnje modelov v fazi modeliranja. Nato so opisani algoritmi uporabljeni za izgradnjo napovednih modelov ter način testiranja modelov. Za klasifikacijo so bili uporabljeni naslednji algoritmi: sekvenčna minimalna optimizacija, logistična regresija, naključni gozd, J48 in naivni Bayes. Rezultati testiranja modelov so podrobno opisani v osmem poglavju rezultati in vrednotenje.

2 STROJNO UČENJE IN PODATKOVNO RUDARJENJE

Podatkovno rudarjenje in strojno učenje sta raziskovalna področja računalništva, ki doživljata hiter razvoj zaradi napredka v raziskovanju analize podatkov in rasti podatkovnih baz uporabljenih v industriji ter posledične potrebe trga po novih metodah za pridobivanje dragocenih znanj oziroma informacij iz velikih zbirkah podatkov. Pojma podatkovno rudarjenje in strojno učenje sta tesno povezana, saj v procesu podatkovnega rudarjenja se pogosto uporablja tehnike strojnega učenja [8].
Podatkovno rudarjenje ni novo odkritje digitalne dobe ampak je koncept, ki se razvija že celo stoletje. Thomas Bayes je leta 1763 objavil članek glede izreka, ki je osnova za podatkovno rudarjenje, kot za teorijo verjetnosti imenovan Bayesov izrek [9]. Razvoj je pospešil leta 1936, ko je angleški matematik Alan Turing uvedel pojem Turingovega stroja [10], ki je sposoben računati in opravljati operacije, kot jih opravljajo sodobni računalniki. Današnji računalniki temeljijo na konceptih, ki je nastavil Alan Turing. Že leta 1943 Warren McCollouch in Walter Pitts so prvi uvedli koncept nevronske mreže v članku [11]. Opisali so nevron v nevronski mreži in določili, katere operacije lahko izvede: sprejemi vhod, procesiraj vhod in generiraj izhod. Komaj leta 1990 je bil prvič uveden izraz podatkovno rudarjenje (Data Mining) [12], in od takrat se tehnike podatkovnega rudarjenja in strojnega učenja hitro razvijajo ter so uporabljene v podjetjih in finančnih skupnosti za analizo podatkov, prepoznavanje trendov, privlačiti nove stranke, napovedovanje nihanj obrestnih mer, napovedovanje cen delnic in povpraševanja kupcev.

2.1 Strojno učenje

Strojno učenje je lahko mišljeno, kot samodejno učenje oziroma sposobnost nekega stroja (računalnika), da se nauči nekih pravil brez da bi bila ta pravila predčasno programirana.
Torej strojno učenje omogoči da se računalnik sam nauči iz izkušenj iz preteklosti, oziroma nek program, ki ima vgrajene algoritme strojnega učenja se izvaja nad podatki in se nauči neka pravila ter samodejno izboljša svoje delovanje med časom izvajanja. Postopek strojnega učenja poteka samodejno brez interakcije uporabnika oziroma uporabnik prepusti računalniku da sam reši problem, brez da bi uporabnik programiral ustrezno rešitev.
Izraz strojno učenje je bil prvič uporabljen v članku, ki je objavil Ameriški pionir v področju umetne inteligence Arthur Lee Samuel leta 1959 [13]. Najbolj uporabljeno definicijo pa jo je izrazil Tom M. Mitchell in pravi, da nek program se uči iz izkušnje E na podlagi neke naloge T z meritvijo učinkovitosti P, če učinkovitost P pri izvajanju naloge T narašča z izkušnjo E [14].

Tehnike strojnega učenja so pogosto uporabljene v procesu podatkovnega rudarjenja, ker omogočajo da se računalnik samodejno nauči iz podatkov brez potrebe uporabnikove interakcije. Ker v današnjem času se proizvede vsak dan ogromno količino podatkov, je koristno analizirati te podatke in poiskati oziroma izluščiti informacije, ki ti podatki prenašajo. Ker podatkov je ogromno oziroma niso popolni ali vsebujejo anomalije, postopek črpanja informacij iz podatkov je zelo naporna naloga za človeka. Tukaj nastopijo v korist računalniki zmožni učenja oziroma programi, ki so zmožni sami identificirati vzorce v podatkih in odkriti neke smiselne povezave za izboljšati njihovo delovanje oziroma nuditi neke informacije človeku iz velikih količin podatkov.

2.1.1 Nadzorovano in nenadzorovano učenje

Strojno učenje raziskuje kako se lahko računalnik nauči oziroma izboljša svoje delovanje na podlagi podatkov. Učenje računalnika preko obdelave podatkov lahko poteka na dva načina:

Nadzorovano učenje: Računalnik se uči na podlagi primerov, za katere se že pozna pripadajoči razred, ta množica podatkov je imenovana učna množica in vsebuje atribut razred, ki točno določa razred objekta. Običajno nadzorovano učenje se uporablja za klasifikacijo oziroma regresijo (več o klasifikacij in regresiji v odstavku 2.21) in deluje tako da računalnik se uči na podlagi učne množice in nato se skuša napovedati razred podatkov iz testne množice. Testna množica vsebuje podobne podatke, kot učna množica ampak brez razrednega atributa, vrednost razrednega atributa je samodejno napovedana od računalnika na podlagi izkušnje pridobljene od učne množice.

Nenadzorovano učenje: Proces učenja je nenadzorovan, ker učni podatki ne vsebujejo razrednega atributa. Tipična aplikacija nenadzorovanega učenja je razvrščanje v skupine, kjer se skuša razvrstiti podatke v skupine oziroma razrede (več o razvrščanju v skupine v odstavku 2.21) . Vhodni podatki za nenadzorovano učenje so lahko množica slik ročno napisanih številk od 0 do 9. Predpostavimo da z uporabo razvrščanja v skupine pridobimo 10 različnih skupin. Te skupine lahko pripadajo desetim različnim številk od 0 do 9 ampak, ker učna množica nima definiranega razrednega atributa, zgrajen model nam ne more izraziti semantičnega pomena najdenih skupin.

Aktivno učenje: je pristop strojnega učenja, kjer uporabnik sodeluje pri procesu učenja. Pri aktivnem učenju je zahtevano sodelovanje uporabnika za označiti primere iz neke množice neoznačenih primerov, ki jih izbere učni program [8].

2.2 Podatkovno rudarjenje

Podatkovno rudarjenje je pogosto opredeljeno, kot avtomatsko pridobivanje novih in zanimivih informacij iz ogromnih količin podatkov. Podatkovno rudarjenje, kot ga trenutno poznamo, ima svoje korenine v statistiki, teoriji verjetnosti in umetni inteligenci. Trenutna popularnost izraza podatkovno rudarjenje se lahko pripiše vzponu KDD konferenci (Knowledge Discovery in Databases and Data Mining Conference) [15]. KDD konferenca se je začela v zgodnjih devedesetih, kot majhna delavnica, ki jo je vodila Usuama Fayyad, Gregory Pietatetsky Shapiro ter Ramasamy Uthurusamy in sedaj konferenca se izvaja letno, z veliko prisotnostjo. V objavljenem članku po KDD konferenci v Montreal leta 1995 [16], podatkovno rudarjenje je bilo definirano, kot: proces samodejnega pridobivanja veljavnih, novih, potencialno uporabnih in razumljivih informacij iz velikih podatkovnih baz.
Podatkovno rudarjenje ni individualna disciplina, ampak zahteva kombinacijo različnih disciplin in tehnik, ter se razlikuje od tradicionalne statistične analize podatkov, ker se obravnava zelo velike nabore podatkov od nekaj gigabajtov, terabajtov do petabajtov. Za nekaj časa se je obravnavalo problem kot problem podatkovnega rudarjenja le, če podatkovna zbirka je bila tako široka, da ni bilo mogoče jo shraniti v celoti v delovni pomnilnik računalnika.
Na splošno lahko povzamemo, da tipičen problem podatkovnega rudarjenja vsebuje naslednje lastnosti: [15]

- Široko zbirko podatkov
- Problem je zahteven in pogostoma slabo opredeljen
- Podatkovna zbirka vsebuje manjkajoče ali napačne podatke
- Podatkovna zbirka vsebuje redundance

Prve zelo uspešne aplikacije podatkovnega rudarjenja so se izvedle na področju trženja oziroma oglaševanja akcijah v supermarketih in načrtane umestitve izdelkov v supermarketih za ojačiti prodajo bližnjih izdelkov. Aplikacija podatkovnega rudarjenja na primer v supermarketu je razkrila, da se prodaja piva povečuje, ko je pivo postavljeno blizu plenic, ker je zelo verjetno da oče kupi tudi paket piva, ko žena ga pošlje v trgovino kupiti plenice. Podatkovno rudarjenje je bilo uspešno aplicirano tudi v področju zlorabe bančnih kartic, vzpostavitev posojil in zlorabe preko telefona pa še na veliko različnih področji npr. v geologiji za iskanje ugodnih lokacij za črpanje nafte ali zemeljskega plina, v kmetijstvu za napovedovanje problemov pri fermentaciji vina ter v bioinformatiki za diagnosticiranje bolezni.
Podatkovno rudarjenje je interdisciplinarna znanost, ki se širi od domene statistike do obdelave informacij, sistemi podatkovnih baz, strojno učenje in umetne inteligence.

Podatkovno rudarjenje ne sloni le na gradnji modelov ali dobrih klasifikatorjev podatkov ampak se skuša pridobiti novo ali globlje razumevanje problemov.
Vodilna misel pri podatkovnem rudarjenju je, da podatke gledamo na drugačen način in v določenem pomenu ter omogočimo podatkom da se sami opišejo. [15]

2.2.1 Metode podatkovnega rudarjenja

Podatkovno rudarjenje je proces, ki vključuje zbiranje podatkov, čiščenje podatkov, pred obdelava in pretvorba podatkov v podatkovno datoteko, gradnja enega ali več modelov oziroma razvrščanje v skupine ali vizualizacija podatkov, ki vodi k oblikovanju pravil za rešiti problem podatkovnega rudarjenja. Cilj procesa podatkovnega rudarjenja je napovedovanje oziroma odkritje nekih pravil ali vzorcev v podatkih, ki nam razkrijejo skrito informacijo v ogromni količini podatkov.
Za odkriti čim večje znanje iz podatkov je potrebno uporabiti različne tehnike podatkovnega rudarjenja, pri čemer je potrebno paziti da se uporabi ustrezno tehniko primerno za določen problem. [8]

2.2.1.1 Pogosti vzorci in asociacijska pravila

V velikih zbirkah podatkov se lahko pojavijo različni tipi vzorcev npr. pogosta množica postavk, pogosta pod zaporedja in pogoste podstrukture.

Pogosta množica postavk se običajno nanaša na niz elementov, ki se pogostoma pojavijo skupaj v transakciji, na primer mleko in kruh se pogosto kupijo skupaj v trgovini.
Z vzorcem pogosta množica postavk skušamo napovedati katere izdelke bo še kupil kupec če vemo da ga zanima nek točno določen izdelek.
Pogosta pod zaporedja so vzorec, ki prikaže kaj je kupec kupil v časovnem zaporedju npr. kupec je najprej kupil prenosni računalnik, potem digitalni fotoaparat in nato pomnilniško kartico. S tem vzorcem skušamo napovedati kaj potrebuje kupec oziroma kateri bo svoj naslednji nakup.
Pogoste podstrukture se lahko nanašajo na različne strukturne oblike (npr. grafi ali drevesa), ki so lahko združene z pogosto množico postavk ali pogostimi pod zaporedji.
Iskanje pogostih struktur lahko pripelje do zanimivih asociacij in korelacij v podatkih.

Asociacijska analiza se izvaja nad neko množico transakcij v kateri se išče pravila, ki bodo napovedala pojavitev nekega predmeta na osnovi ostalih predmetov v transakciji. Na primer

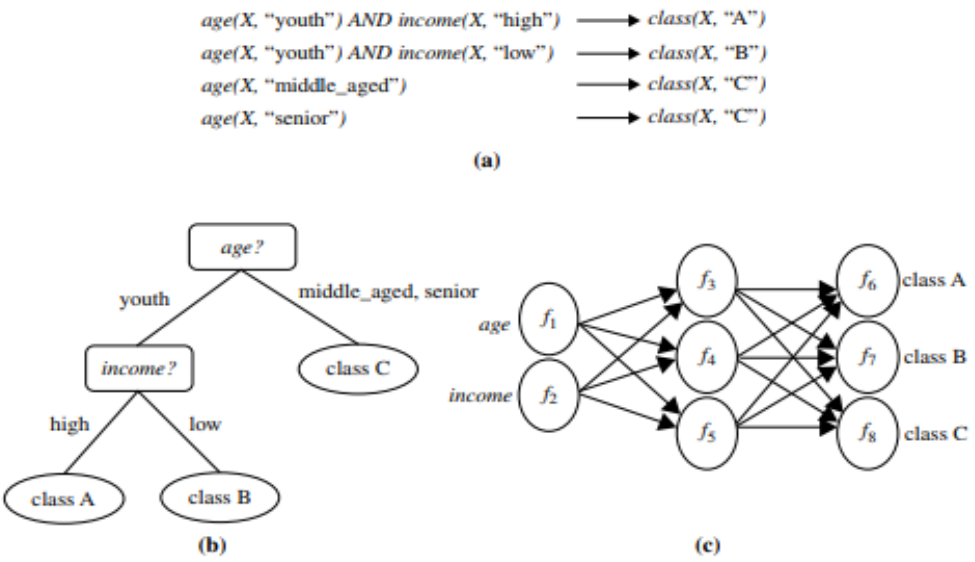
manager neke računalniške trgovine želi zvedeti kateri predmeti so pogostoma kupljeni skupaj (v isti transakciji). Primer takega pravila je naslednji:

kupil(X , računalnik) ⇒ kupil(X , programska oprema)[support=1% , confidence=50%]

Spremenljivka X predstavlja kupca.
Confidence ali certainty oziroma zaupanje 50% označi da če kupec X kupi računalnik potem je 50% možnosti da bo kupil tudi programsko opremo.
Support ali podpora 1% pomeni da iz vseh analiziranih transakcij 1% transakcij vsebuje računalnik in programsko opremo skupaj.
To je bil primer eno-dimenzijskega asociacijskega pravila, ker upoštevamo le vrednosti enega atributa, asociacijsko analizo pa lahko izvajamo tudi nad več atributi in tako razkrijemo več-dimenzionalna asociacijska pravila. [8]

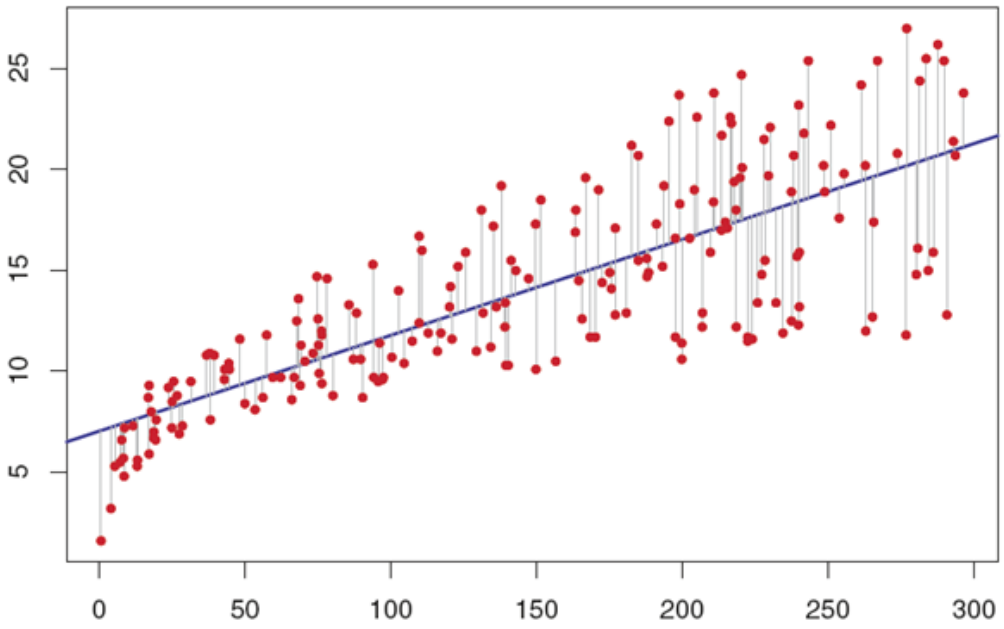
2.2.1.2 Klasifikacija in regresija

Klasifikacija je proces iskanja modela ali funkcije, ki opiše in razdeli podatke v vnaprej definirane razrede ali koncepte [8]. Model je izpeljan iz analize učne množice. Učna množica je množica podatkov, ki ima že opredeljen pripadajoči razred in se uporabi za zgraditi model ter ga oceniti. Nato zgrajen model se aplicira nad podatki za katere se ne pozna pripadajoči razred ter se ga poskusi napovedati.
Zgrajen model je lahko predstavljen v različnih oblikah, kot klasifikacijska pravila, odločitvena drevesa, matematične formule ali nevronske mreže.



Slika 1: Klasifikacijski model je lahko predstavljen na več različnih načinov: (a) klasifikacijska pravila, (b) odločitveno drevo, (c) nevronska mreža (vir: [8])

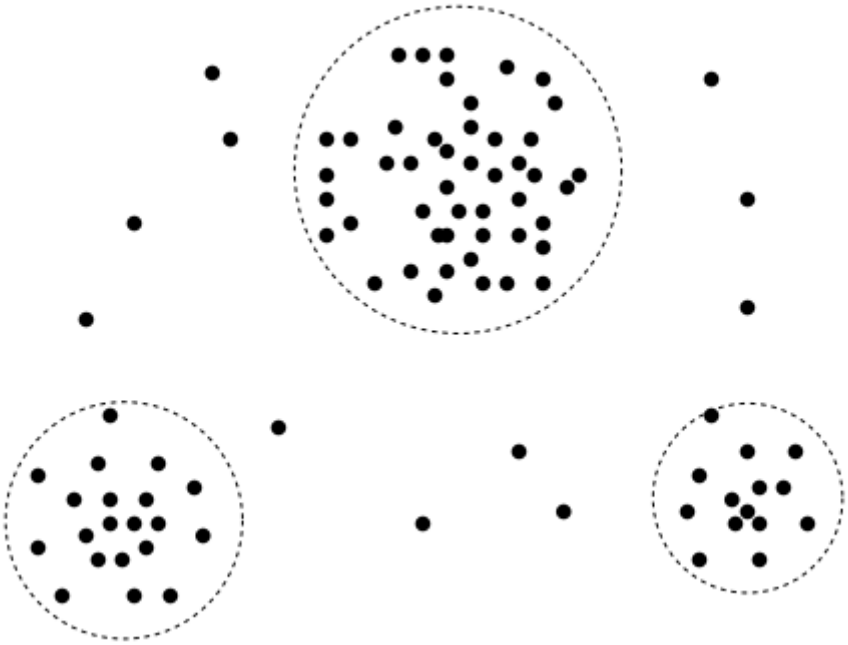
Klasifikacija je uporabljena za napovedati diskretne razrede podatkov, medtem ko **regresija** je uporabljena za opisati in razdeli podatke v numerične razrede.
Diskretni podatki so podatki, ki zavzamejo končno število različnih vrednosti. Numerični podatki pa so zvezni podatki in lahko zavzamejo neskončno število različnih vrednosti.
Na primer regresija bi bila uporabljena za napovedati vrednost neke nepremičnine na podlagi svoje lokacije, površine, cene prodaje, cene hiš v okolici in ostalih faktorjev.
Najpreprostejša oblika regresije je linearna regresija, ki meri odvisnost dveh spremenljivk oziroma kako ena spremenljivka vpliva na drugo. Ta tehnika uporablja matematično formulo premice (y = mx + b). To pomeni da če imamo nek graf z X in Y osjo, potem odvisnost med X in Y je premica, to zahteva strogo linearno odvisnost med podatki. Na primer če si predstavljamo nek graf kjer Y os predstavlja povečanje populacije in X os predstavlja proizvodnjo hrane. Kot bo naraščala vrednost Y v grafu bo tudi vrednost X linearno naraščala in odvisnost med X in Y osjo bo premica [8].



Slika 2: Primer premice linearne regresije (vir: [17])

2.2.1.3 Razvrščanje v skupine

Razvrščanje v skupine deluje na različen način od klasifikacije in regresije, saj se ne gradi model na podlagi učne množice, ki ima že definirane razrede ampak se analizira podatkovne objekte brez danega razreda. V veliko primerih na začetku procesa podatkovnega rudarjenja so razredi podatkov neznani in s pomočjo razvrščanja v skupine skušamo opredelit razrede ter definirati pravila, ki bodo določila razred podatkom.

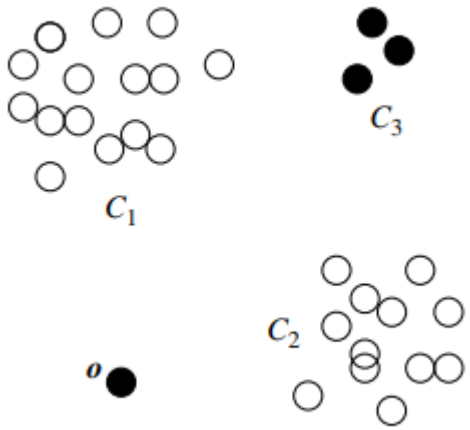


Slika 3: 2-D prikaz treh skupin podatkov ob postopku razvrščanja v skupine (vir: [8])

Razvrščanje v skupine je lahko uporabljeno za generirati razredni atribut za neko množico podatkov. Objekti so razvrščeni v skupine za doseči čim višjo podobnost med podatki znotraj razreda in čim nižjo podobnost med podatki različnih razredov.
Vsaka ustvarjena skupina je lahko gledana kot nek razred objektov, iz katerih je mogoče izpeljati pravila za gradnjo modelov [8].

2.2.1.4 Analiza anomalij

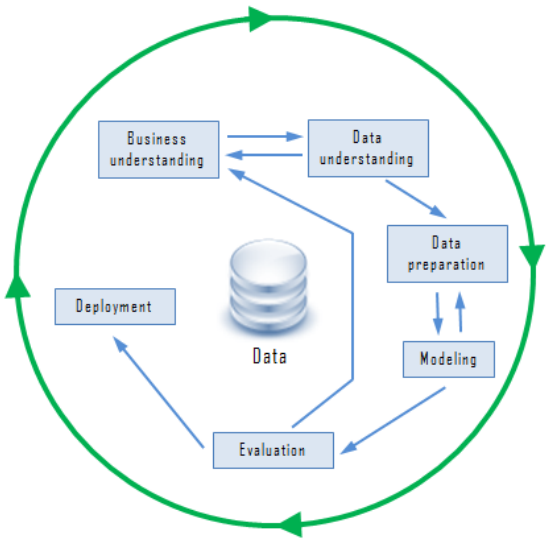
Neka podatkovna množica lahko vsebuje objekte, ki ne ustrezajo generalnemu modelu podatkov ampak izpadajo iz njega. Ti podatkovni objekti so imenovani anomalije. Veliko metod podatkovnega rudarjenja upošteva anomalije, kot nek šum oziroma izjema v podatkih ter jih zavrže pri gradnji končnega modela.
Vendar v nekaterih primerih podatkovnega rudarjenja (npr. zloraba bančnih kartic) izjemni dogodki so lahko zanimivejši od rednih običajnih dogodkov.
Analiza anomalij je zmožna odkriti zlorabo bančnih kartic, tako da se zazna nenavadno visoke zneske za določeno številko računa v primerjavi z navadnimi rednimi zneski istega računa. Anomalije se lahko tudi zazna iz lokacije nakupa, tipa nakupa ali frekvence nakupov. Statistične metode so uporabljene v analizi anomalij z upoštevanjem verjetnostne porazdelitve modela podatkov kot tudi merjenje razdalje z uporabo razvrščanja v skupine, kjer objekti oddaljeni od katerekoli skupine so obravnavani kot anomalije [8].



Slika 4: Analiza anomalij z razvrščanjem v skupine, C1,C2,C3 so skupine podatkov, O je zaznana anomalija (vir: [8])

2.3 CRISP-DM metodologija

CRISP-DM je kratica za **Cr**oss-**I**ndustry **S**tandard **P**rocess for **D**ata **M**ining oziroma Industrijski standard, ki natančno določa potek analize podatkov s pomočjo podatkovnega rudarjenja. Proces podatkovnega rudarjenja je potrebno standardizirati, ker mora biti zanesljiv in ponovljiv.
CRISP-DM je bil zasnovan leta 1996 v še mladem in nerazvitem trgu podatkovnega rudarjenja [4]. Standard ni lastniški, je neodvisen od aplikacije oziroma problema, neodvisen od programskega orodja, se osredotoči na poslovni problem in na tehnični vidik analize ter služi kot ogrodje, ki nas vodi preko celotnega procesa.
Metodologija razdeli projekt podatkovnega rudarjenja v šest faz in omogoča premikanje v naprej kot tudi v nazaj med različnimi fazami [4].



Slika 5: Prikaz faz CRISP-DM metodologije (vir: [18])

Faze CRISP-DM metodologije:

1. **Razumevanje problema**

Potrebno je razumeti cilje in potrebe projekta, ter definirati problem podatkovnega rudarjenja in načrtati plan izvajanja projekta.
Prvi korak, ki se izvede ob začetku kateregakoli projekta je ugotoviti in natančno definirati cilje projekta. Potrebno je začeti projekt z popolno jasnost problema in cilja projekta, ter je potrebno točno določiti omejitve in vpliv, ki bo imel projekt na poslovanje (organizacijo).
Potrebno je definirati cilje podatkovnega rudarjenja npr. modele, poročila, predstavitve itd.
Nato se izdela projektni plan, ki vsebuje vse korake potrebne za dokončat in predstavit projekt.

2. Razumevanje podatkov

Najprej je potrebno podatke zbrati in preveriti, da ustrezajo našim potrebam. To je zelo važen korak procesa podatkovnega rudarjenja, ker vsak postopek procesa temelji na tem da so na volijo pravilni podatki in brez pravilnih podatkov ni moč doseči končnega cilja. Sedaj je potrebno temeljito opisati zbrane podatke oziroma opisati vir podatkov, format podatkov, število zbranih primerov itd.
Nato je potrebno temeljito raziskati podatke za vsak atribut posebej ter aplicirati statistične metode za analizo. Potrebno je oceniti kakovost podatkov in določiti ali podatki ustrezajo našim potrebam za doseči predpostavljen cilj.

3. Priprava podatkov

Potrebno je izbrati katere podatke se bo uporabilo v procesu podatkovnega rudarjenja, nato se prečisti podatke tako da odstranimo vse napake v podatkih. Sledi sestava podatkov, kjer se generira nove atribute na podlagi že obstoječih atributov.
Podatke je potrebno združiti v skupno množico podatkov ta postopek se imenuje integracija podatkov. Naslednji korak je formatiranje podatkov, kjer se pretvori format podatkov v format, ki ustreza programski opremi.

4. Modeliranje

Korak modeliranja se začne z selekcijo metode podatkovnega rudarjenja, nato je potrebno načrtati potek testiranja za čim natančneje oceniti učinkovitost zgrajenih modelov.
Sledi gradnja modelov, kjer je potrebno zgraditi modele ter nastavljati parametre metod za zgraditi čim natančnejši model. Nato je potrebno oceniti kakovost modela.

5. Evalvacija

V prvih štirih fazah CRISP-DM procesa je bilo potrebno raziskovati podatke in iskati vzorce v njih, sedaj je potrebno oceniti zgrajene modele in celotno izvajanje procesa podatkovnega rudarjenja ter oceniti praktični potencial zgrajenih modelov.

6. Predaja

Predaja je končna faza CRISP-DM metodologije in vključuje implementacijo modela z celotno dokumentacijo, tako da je celoten proces ponovljiv in razumljiv, planiranje predaje, spremljanje vzdrževanja, končno poročilo in oceno uspeha celotnega projekta [4].

Proces podatkovnega rudarjenja opisan v tej nalogi bo izveden na podlagi CRISP-DM metodologije. Ta metodologija je bila izbrana, ker nudi dober pregled skozi celoten razvoj projekta in vključuje vse faze razvoja projekta od razumevanja problema do predaje v uporabo. CRISP-DM je zelo fleksibilna metodologija, ki se dobro prilagodi na probleme iz

različnih področji in zagotovi da proces podatkovnega rudarjenja je dobro dokumentiran, zanesljiv in ponovljiv tudi s strani ljudi z omejenim znanjem podatkovnega rudarjenja.

3 UPORABLJENA PROGRAMSKA OPREMA

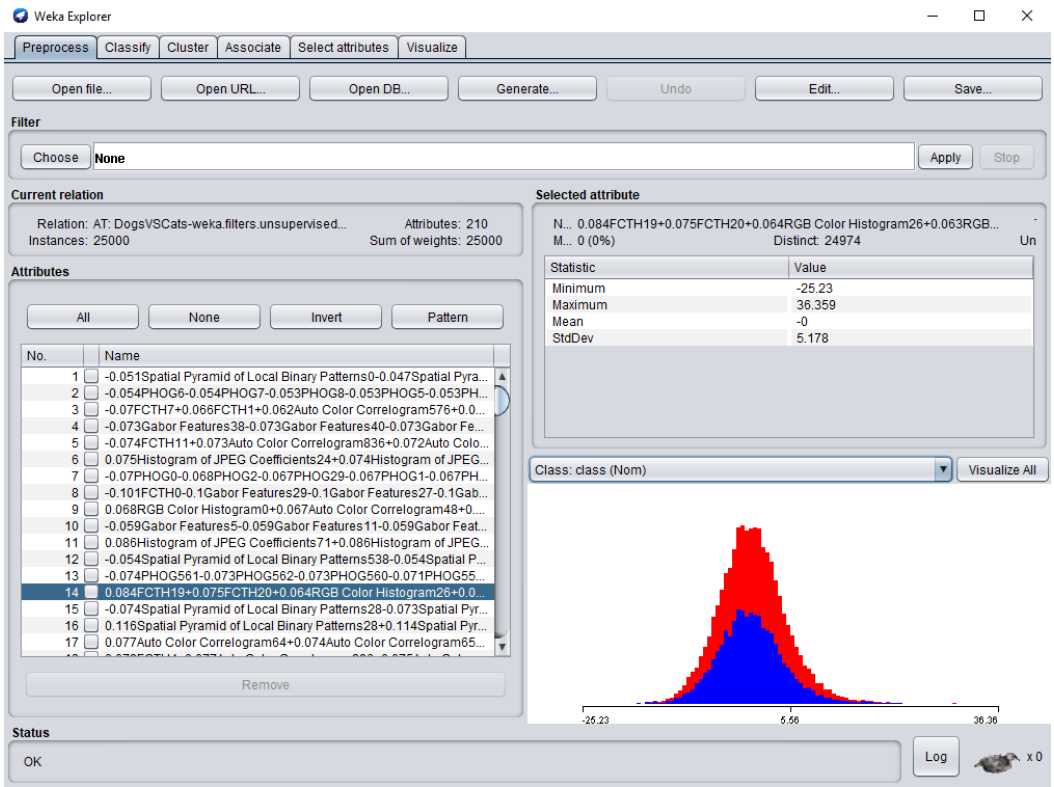
Za izdelavo zaključne naloge je bil uporabljen program za podatkovno rudarjenje Weka ter vtičnik ImageFilter, ki razširja funkcionalnosti programa Weka. Program se izvaja na računalniku z operacijskim sistemom Windows 10, ter za vizualizacijo in pregled podatkov v ARFF formatu je bil uporabljen program Sublime Text 3. Za pretvorbo podatkov v ARFF format pa je bila uporabljena bash skripta na računalniku z operacijskim sistemom Ubuntu.

3.1 Program Weka

Weka oziroma **W**aikato **E**nvironment for **K**nowledge **A**nalysis je zbirka programske opreme za strojno učenje in podatkovno rudarjenje, napisana je v Javi [19] in je bila razvita na Univerzi v Waikatu v Novi Zelandiji. To je brezplačna programska oprema licencirana pod GNU-GPL licenco [20].
Weka vsebuje veliko zbirko orodji za podatkovno rudarjenje oziroma vsebuje orodja za vizualizacijo podatkov, predprocesiranje podatkov, razvrščanje v skupine, klasifikacijo in regresijo. Vse vgrajene metode v Weki temeljijo na predpostavki, da so podatki na voljo v nepovezani datoteki (flat-file) ali relaciji, kjer vsak podatkovni objekt je opisan z fiksnim številom atributov.
Program vsebuje mnogo različnih grafičnih uporabniških vmesnikov, ki omogočijo enostaven dostop do orodji za strojno učenje in podatkovno rudarjenje. Sledi opis uporabniških vmesnikov uporabljenih za razvoj zaključne naloge [21].

3.1.1 Raziskovalec

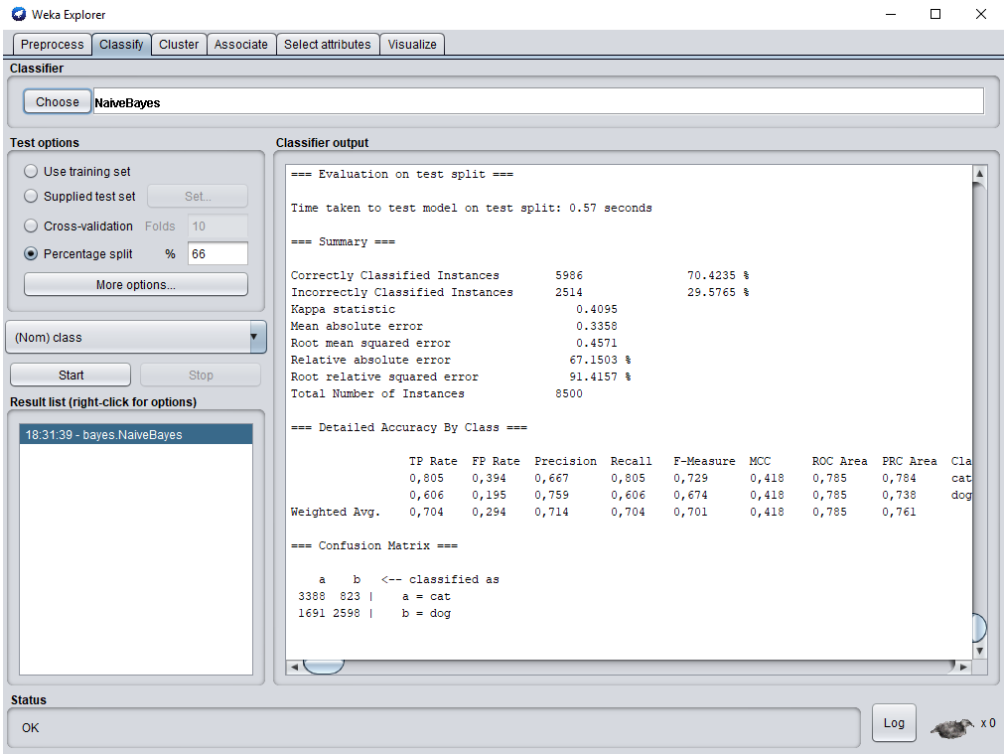
Glavni grafični uporabniški vmesnik v programu Weka je **raziskovalec** (v angl. Explorer). Raziskovalec ima vmesnik zasnovan na več zavihkov, kjer različni zavihki ustrezajo različnim nalogam procesa podatkovnega rudarjenja. V prvem zavihku imenovan Preprocess je mogoče naložiti podatke v ARFF formatu ali preko povezave na podatkovno bazo. Ko se podatki naložijo, se prikažejo tudi osnovne informacije o podatkih npr. ime relacije, število primerov, število atributov za vsak primer ter statistični podatki za posamezni atribut. Na zaslonu je prikazan tudi nek osnovni graf, ki prikazuje razrede primerov v odvisnosti od vrednosti danega atributa. V tem oknu je mogoče podatke tudi prečistiti z uporabo mnogih filtrov, ki so na razpolago pod opcijo filter. Za izdelavo zaključne naloge je ključnega pomena filtriranje podatkov, ker bomo uporabili vtičnik ImageFilter za pretvoriti slike iz JPG formata v kolekcijo atributov, ki bodo naknadno obdelani v programu Weka.



Slika 6: raziskovalec v Weki (zavihek Preprocess)

Zavihek Preprocess v raziskovalcu je najbolj uporabljen v prvih fazah procesa podatkovnega rudarjenja, kjer raziskujemo podatke in skušamo razumeti na kateri način bo najbolje analizirati in zgraditi model nad temi podatki. Gradnja modela pa poteka v zavihku Classify, kjer so nameščena orodja za klasifikacijo in regresijo. V tem zavihku je mogoče izbrat željeni algoritem za klasifikacijo, izbrati način izvajanja testa in razredni atribut za nadzorovano

učenje. Nato je zadosti pritisniti na gumb start za začeti gradnjo modela na podlagi učne množice in po določenem času se prikažejo rezultati gradnje in testiranja modela.



Slika 7: zavihek Classify v raziskovalcu na programu Weka

Za izdelavo zaključne naloge je uporabljen tudi zavihek Select attributes v raziskovalcu, ta zavihek omogoči uporabniku, da iz večjega nabora atributov se analizira in izvleče manjšo množico pomembnejših atributov z uporabo algoritmov in metod za razvrstitev atributov. Zavihek Select attributes omogoči izbiro algoritma, ki bo evalviral atribute in izbiro metode, ki bo razvrstila atribute na podlagi rezultatov prej uporabljenega algoritma. Rezultati so prikazani na zaslonu, ter obstaja možnost za shraniti novo datoteko sestavljeno le iz najboljših atributov.

3.1.2 Preizkuševalec

Preizkuševalec v Weki je namenjen testiranju algoritmov za klasifikacijo in regresijo.
V zavihku Setup se določi podatkovno zbirko nad katero se bo izvajal algoritem, algoritem ter parametre izvajanja ter število ponovitev in način testiranja. Požene se preizkuševalec in se pusti da računalnik samostojno izvaja algoritme za dlje časa in ko vsi algoritmi so prenehali izvajanje se primerja rezultate v zavihku Analize, kjer z uporabo različnih testnih metod se išče najbolj učinkovit algoritem za naš primer podatkovnega rudarjenja.

3.1.3 ARFF format podatkov

ARFF (angl. **A**ttribute-**R**elation **F**ile **F**ormat) je tekstovna ASCII datoteka, ki opiše seznam instanc z množico atributov. ARFF format je bil razvit v Machine Learning Project v oddelku za Računalništvo in Informatiko na Univerzi v Waikato [6] za analizo podatkov in aplikacijo metod podatkovnega rudarjenja v programu Weka.
ARFF datoteka je sestavljena iz dveh različnih sekcij, prva sekcija imenovana Glava (angl. Header) vsebuje ime relacije ter seznam vseh atributov z imenom in tipom. V glavi avtor lahko vpiše ime podatkovne zbirke, ime avtorja ter dodatne informacije o podatkovni zbirki v obliki komentarja. Deklaracija atributov v glavi zgleda, kot urejeno zaporedje atributov, kjer vsak atribut ima prirejeno ime in podatkovni tip. Ter vrstni red atributa v glavi enolično določa pozicijo atributa v podatkovni sekciji ARFF formata. Na primer če nek atribut je deklariran na četrtek mestu, potem Weka pričakuje da vse vrednosti tega atributa so na četrti koloni vsakega podatkovnega objekta. Podatkovni tipi ARFF formata so lahko: numerične oziroma zvezne vrednosti, diskretne vrednosti, niz ali datum.

```
% 1. Title: Iris Plants Database
%
% 2. Sources:
%      (a) Creator: R.A. Fisher
%      (b) Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
%      (c) Date: July, 1988
%
@RELATION iris

@ATTRIBUTE sepallength  NUMERIC
@ATTRIBUTE sepalwidth   NUMERIC
@ATTRIBUTE petallength  NUMERIC
@ATTRIBUTE petalwidth   NUMERIC
@ATTRIBUTE class        {Iris-setosa,Iris-versicolor,Iris-virginica}
```

Slika 8: Glava ARFF formata (vir: [22])

V sliki je prikazana glava ARFF datoteke za podatkovno zbirko Iris Plants Database, kot vidimo glava se začne z kratkim opisom podatkovne zbirke v obliki komentarja. Nato sledi ime relacije: iris in seznam atributov ter zadnji definiran atribut je razredni atribut imenovan class, ki napove razred pripadajočega podatkovnega objekta.

```
@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
```

Slika 9: Podatkovni del ARFF datoteke (vir: [22])

Druga sekcija ARFF datoteke je podatkovni del (angl. Data), v katerem so vsebovani vsi podatkovni objekti podatkovne zbirke. Vsak podatkovni objekt je predstavljen z eno samo vrstico, ki vsebuje vse atribute opisane v glavi datoteke, ter atributi so omejeni z vejico in morajo biti uvrščeni po istem vrstnem redu definiranem v glavi. [6]

3.2 ImageFilter

ImageFilter je vtičnik, ki razširja funkcionalnosti programa Weka z paketom, ki vsebuje filtre za procesiranje slik [7]. Vtičnik je dostopen iz Raziskovalca v programu Weka v zavihku Preprocess. Ko smo naložili ARFF datoteko v zavihku Preprocess je mogoče aplicirati različne filtre. Filtri vtičnika ImageFilter so dostopni preko:

weka -> filters -> unsupervised -> instance -> imagefilter

Image Filter vsebuje deset različnih filtrov, ki procesirajo sliko na različne načine.
Aplikacija filtra deluje tako da se izbere željen filter, ter se ga aplicira nad podatkovno zbirko. Filter obdela vsako sliko posamezno in jo pretvori v množico atributov, ki je shranjena v novo ARFF datoteko [23]. Proizvedena množica atributov se razlikuje za vsak posamezen filter in aplikacija različnih filtrov proizvede različne rezultate. Za doseči čim boljše rezultate v naslednjih fazah podatkovnega rudarjenja je potrebno izbrati najučinkovitejši filter, da proizvede čim boljšo množico atributov [24].

4 RAZUMEVANJE PROBLEMA

V zaključni nalogi je temeljito opisan primer uporabe podatkovnega rudarjenja za detekcijo mačk in psov v slikah. Problem je črpan iz spletne strani https://www.kaggle.com [25], na tej spletni strani je prisotnih veliko zbirk podatkov in problemov, ki zahtevajo uporabo naprednih tehnik podatkovnega rudarjenja in strojnega učenja. Izbran problem se imenuje Dogs vs. Cats [26] in zahteva uporabo tehnik podatkovnega rudarjenja in strojnega učenja za detekcijo mačk in psov v slikah, ter klasifikacijo slike kot mačka, če slika prikazuje mačko in klasifikacijo slike kot pes, če slika prikazuje psa. Problem je bil objavljen 25.09.2013 v obliki tekmovanja, katerega se je udeležilo 250 tekmovalcev.

Postopek razlikovanja je zelo enostaven za človeka, pse in mačke ampak je zahteven izziv za računalnike [26]. Podatki so del večje podatkovne zbirke imenovane Asirra [2] uporabljene za CAPTCHA (angl. **C**ompletely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part) . CAPTCHA je popolnoma avtomatiziran Turingov test s pomočjo katerega ločimo računalnike in ljudi med seboj. Podatkovna zbirka Asirra vsebuje veliko število slik mačk in psov, uporabljenih za izvajanje Turingovega testa nad uporabniki spletnih storitev, za preverjati če so dejansko ljudje, ki dostopajo do storitve. Asirra je edinstvena podatkovna zbirka zaradi partnerstva s Petfinder.com [27], zelo uporabljena spletna stran za iskanje domov domačim živali. Petfinder.com je priskrbel Microsoftu več kot tri milijone slik mačk in psov, ki so jih ročno klasificirali ljudje v zavetiščih za živali v ZDA in tako je nastala podatkovna zbirka Asirra.

Glavni cilj zaključne naloge je zgraditi model, ki omogoči avtomatično detekcijo mačk in psov v slikah z dobro natančnostjo. Sedanja literatura svetuje da z uporabo naprednih metod podatkovnega rudarjenja je mogoče zgraditi model, ki je zmožen doseči čez 80% točnosti nad to nalogo [2]. Zato Asirra ni več varna pred napadom novejših tehnik strojnega učenja in podatkovnega rudarjenja.

5 RAZUMEVANJE PODATKOV

V drugi fazi CRISP-DM metodologije je potrebno zbrat podatke potrebne za izvest proces podatkovnega rudarjenja ter jih temeljito raziskovat in preveriti da ustrezajo našim potrebam.

5.1 Zbiranje podatkov

Podatki so črpani iz spletne strani Kaggle [25] in so del tekmovanja Dogs vs. Cats [26]. Kot opisano v prejšnjem odstavku podatki so del večje zbirke imenovane Asirra [2], ki vsebuje več kot tri milijone primerov digitalnih slik mačk in psov.
Podatki uporabljeni za izdelavo zaključne naloge so strukturirani v dveh različnih množicah. V učni množici je 25.000 klasificiranih primerov, med katerimi so 12.500 slik mačk in 12.500 slik psov in v testni množici je 12.500 neklasificiranih slik mačk ali psov.

5.1.1 Pretvorba podatkov v ARFF format

Podatki pridobljeni iz spletne strani Kaggle [25] so komprimirani v zip datoteki in sicer učna množica zaseda 556 MB ter testna množica 277 MB. Komprimirana datoteka učne množice vsebuje 25.000 klasificiranih primerov, kjer vsak primer je oblike digitalne slike JPG formata.
Vsaka slika prikazuje psa ali mačko in ime slike je oblike:

- cat.1000.jpg
- dog.1001.jpg

Torej v učni množici je mogoče identificirati katera žival je na sliki iz imena slike, zato potrebujemo neko skripto, ki pretvori ime slike v atribut ustrezen za ARFF format.
Odločili smo se za izdelavo BASH skripte, ki se sprehodi skozi vse elemente učne množice ter ustvari novo datoteko, v kateri vsak podatkovni element je opisan z imenom slike in razrednim atributom, ki je črpan iz imena slike. Izdelana skripta se izvaja na računalniku z operacijskim sistemom Ubuntu ter pretvori direktorij slik učne množice v datoteko, ki vsebuje naslednje podatke:

```
12519  cat.9998.jpg,cat
12520  cat.9999.jpg,cat
12521  cat.999.jpg,cat
12522  cat.99.jpg,cat
12523  cat.9.jpg,cat
12524  dog.0.jpg,dog
12525  dog.10000.jpg,dog
12526  dog.10001.jpg,dog
12527  dog.10002.jpg,dog
12528  dog.10003.jpg,dog
12529  dog.10004.jpg,dog
```

Slika 10: prikaz podatkov v datoteki po aplikaciji BASH skripte

Iz slike je razvidno da vsaka vrstica predstavlja drugi podatkovni objekt in v vsaki vrstici sta dva atributa omejena z vejico. Prvi atribut predstavlja ime podatkovnega objekta, drugi atribut pa je razredni atribut in napove razred podatkovnega objekta. Da lahko Weka ustrezno prepozna podatke, je potrebno izdelati glavo ARFF datoteke.

```
1   %   1. Naslov: Uporaba podatkovnega rudarjenja za detekcijo mačk in psov v slikah
2   %   1. Title:  The use of Data Mining Methods to Detect Cats and Dogs in images
3   %
4   %   2. Vir podatkov: https://www.kaggle.com/c/dogs-vs-cats
5   %
6   %   3. Avtor: Niki Hrovatin
7   %
8   %   4. Datum: 10/06/2018
9   %
10  %
11  %
12  % ----> Učna množica
13
14  @RELATION MackeVSPsi
15
16  @ATTRIBUTE filename string
17  @ATTRIBUTE class {cat,dog}
18
19
20
21  @DATA
22  cat.10000.jpg,cat
23  cat.0.jpg,cat
24  cat.10001.jpg,cat
```

Slika 11: ARFF datoteka učne množice

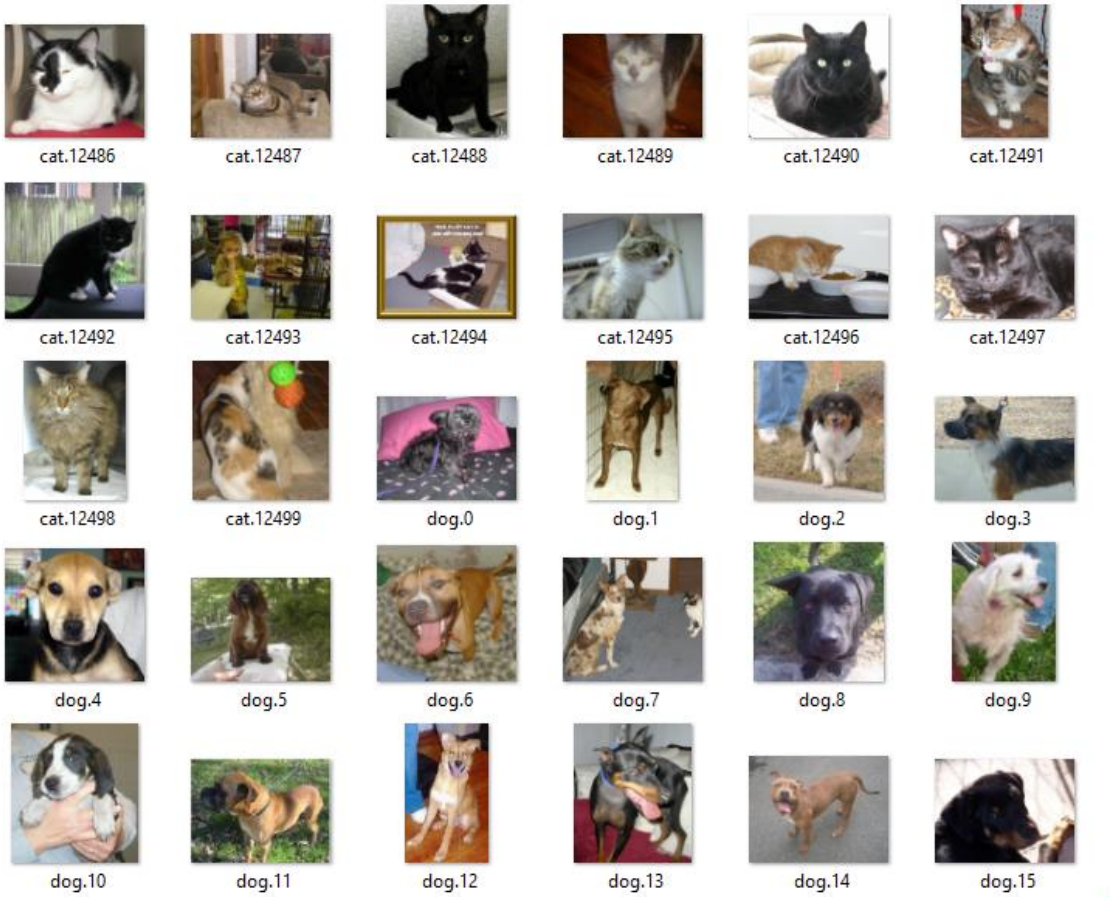
Iz slike je razvidno, da v glavi ARFF datoteke smo dodali kratek opis podatkovne množice, ime relacije, imena atributov ter oznako @DATA, ki nakazuje da od nje dalje se prične podatkovni del ARFF datoteke. Datoteke vsebuje le dva atributa in sicer: filename, ki predstavlja ime datoteke in je tipa string, ta atribut bo uporabljala Weka za dostopati do slike in atribut class, ki napove razred podatkovnega objekta in lahko zavzame le dve vrednosti: cat ali dog.
Sedaj podatki so pripravljeni ter program Weka jih bo ustrezno prepoznal.

5.2 Opis podatkov

Podatkovna zbirka je sestavljena iz naslednjih množic podatkov:

- **Učna množica:** 25.000 slik, od katerih je 12.500 slik mačk in 12.500 slik psov
- **Testna množica:** 12.500 slik mačk ali psov

V učni množici slike so prirejene z razrednim atributom, ki napove pripadajoči razred slike. Ko dekomprimiramo zip datoteko učne množice pridobimo direktorji, ki vsebuje 25.000 slik mačk ali psov. Slike so JPG [28] formata, ki je sinonim za JPEG (angl. **J**oint **P**hotographic **E**xperts **G**roup), to je prvi mednarodni standard za kompresijo digitalne slike, ki sedaj je tudi svetovno najbolj uporabljen. JPEG format slik omogoči da 25.000 slik iz učne množice zasedata le 556 MB. Slike v učni množici so različnih velikosti in prikazujejo živali pred različnimi ozadji, z različnimi objekti, iz različnih kotov, različnih poz , slike so tudi različno osvetljene in si niso podobne.



Slika 12: slika prikazuje podatke iz učne množice

Kot opazimo iz zgornje slike prikazane živali so različne pasme, različnih velikosti, usmerjene v različne smeri in vse slike so precej različne, razlikovanje med psi in mačkami

je enostavna naloga za človeško oko ampak klasifikacija tako raznolikih slik je zahtevna naloga za računalnike.

5.3 Raziskovanje podatkov

V fazi raziskovanja podatkov bo upoštevana le učna množica, ker testna množica je analogna učni množici brez razrednega atributa.
V programu Weka smo odprli uporabniški vmesnik Raziskovalec na zavihku Preprocess ter smo naložili ARFF datoteko učne množice. Na zaslonu se prikažejo osnovni podatki o učni množici:

- Ime relacije: MackeVSPsi
- Število atributov: 2
- Število instanc: 25.000
- Atribut filename tipa string, ki vsebuje ime slike
- Atribut class, ki napove pripadajoči razred slike

Učna množica vsebuje 25.000 slik mačk in psov, od katerih je 12.500 slik psov in 12.500 slik mačk. Digitalne slike so barvne različnih velikosti, oblik in različnih ločljivosti. Ločljivost slik se giblje od 250x250 pixlov do 500x500 pixlov. V naslednjem odstavku bodo opisani opazki nad slikami v učni množici:

- Vsaka slika ima različno ozadje oziroma različne predmete v ozadju npr. ure, kletke, odeje, postelje, osebe itd.
- Živali so v različnih pozah, slikane iz različnih kotov, ne vse živali gledajo v kamero.
- Slike so različno osvetljene.
- Na nekaterih slikah so prikazane dve ali več živali skupaj.
- Ni slik kjer sta prikazani pes in mačka skupaj.
- Živali so različne pasme.
- Živali so različnih starosti, prisotni so tudi mladiči.
- Na nekaterih slikah živali so prikazane z plašči.
- Nekatere slike so preoblikovane.
- Nekatere slike ne prikazujejo živali oziroma prikazujejo ilustrirano žival.

5.4 Preverjanje kakovosti podatkov

Iz zgornjih opazk se takoj ugotovi da imamo opravka s precej zahtevno zbirko podatkov, saj vsebuje veliko problemov, ki ni mogoče rešiti. Menimo da največje težave povzroči to da živali so različne pasme in različnih starosti, ker nekateri mladiči psov so precej podobni mucam in če živali so različne pasme so tudi različnih barv, kar uteži postopek klasifikacije preko tehnik, ki uporabljajo zaznavanje barv. Velike težave povzroči tudi poza živali in kot s katerim so slikane. Ozadje v slikah ni velik problem, saj želimo razlikovat žival od ozadja in ne upoštevati ozadje. Različna osvetlitev slik ne povzroči posebnih težav.
Pojavitev dveh ali več živali na sliki ne povzroča težav, oziroma lahko nam olajša delo, ker imamo več možnosti za pravilno zaznati žival. Živali slikane z plašči ne povzročajo nadlog, ker plašč ne prekriva glave ampak telo in postopek identifikacije živali poteka po večini iz glave.

Slika 13: prikaz slik, ki vsebujejo omenjene težave

V zgornji sliki se takoj opazi 3 primere, ki ne prikazujejo živali oziroma prikazujejo illustrirano žival, načeloma take slike so velik problem, ker so zelo različne od ostalih in za računalnik je skoraj nemogoče jih razlikovat. Take slike bi bilo potrebno ročno odstraniti ampak zaradi obsežne podatkovne množice in izredne redkosti takih slik bomo pustili tudi te primere v učni množici, saj ročna odstranitev bi zahtevala veliko časa. Tudi preoblikovane slike ne predstavljajo velikih težav, saj žival v sliki je poudarjena nad ozadjem.
Podatkovna množica vsebuje kar 25.000 primerov, zato je zadosti obširna da zgoraj omenjeni problemi ne povzročijo neprehodnih težav, ki bi onemogočile izvedbo projekta.

6 PRIPRAVA PODATKOV

Podatki pridobljeni iz spletne strani Kaggle [26] so le zbirka digitalnih slik mačk in psov, ne vsebujejo nobenega atributa na podlagi katerega bi lahko napovedali pripadajoči razred podatkovnega objekta. V fazi priprave podatkov bodo uporabljene napredne tehnike podatkovnega rudarjenja za zajeti lastnosti iz digitalnih slik, ki bodo uporabljene za gradnjo modelov v naslednjih fazah procesa podatkovnega rudarjenja.

6.1 Izbira podatkov

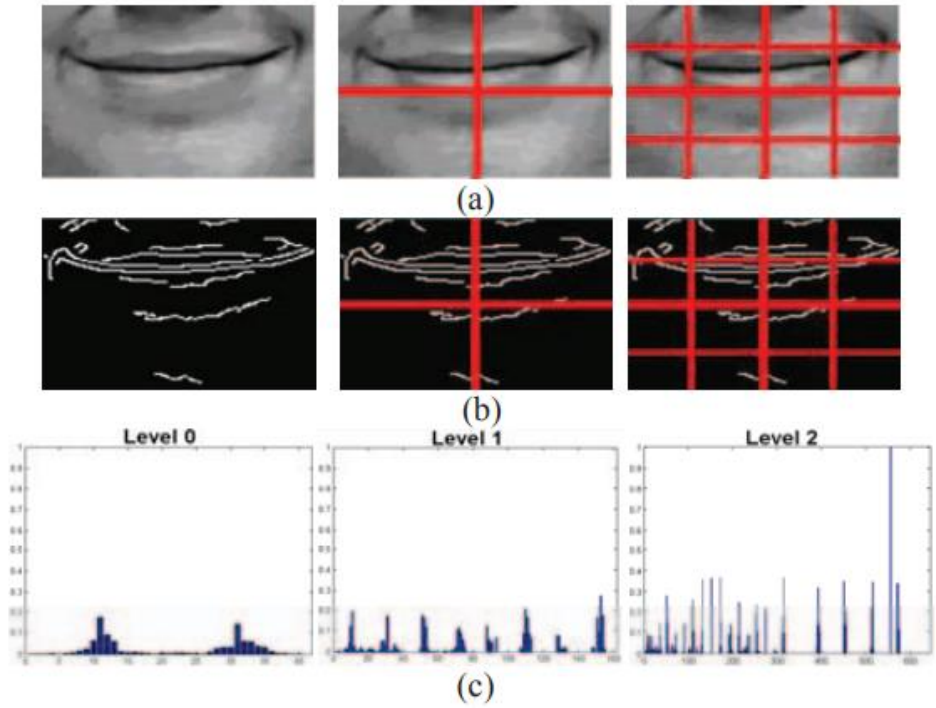
Ker podatkovna zbirka vsebuje veliko primerov, skupno učna in testna množica vsebujeta, kar 37.500 primerov. Za izvedbo procesa podatkovnega rudarjenja bodo uporabljeni le podatki vsebovani v učni množici podatkovne zbirke. Učna množica vsebuje 25.000 primerov, kar je čisto dovolj za opraviti gradnjo in evalvacijo modelov nad isto množico.

6.2 Sestava podatkov

V fazi priprave podatkov procesa podatkovnega rudarjenja je potrebno pripraviti podatke za gradnjo modelov. Ker naša podatkovna zbirka je sestavljena iz množice digitalnih slik je potrebno analizirati slike, ter jih pretvoriti v množico atributov. Za izvesti pretvorbo digitalne slike v množico atributov uporabimo vtičnik **ImageFilter** [7]**.** Vtičnik razširja funkcionalnosti programa Weka z paketom, ki vsebuje deset filtrov za procesiranje slik.
Filtri vsebovani v vtičniku se uporabljajo za slike z različnimi lastnostmi, oziroma nekateri filtri so boljši za prepoznavanje barvnih lastnosti ter nekateri so bolj občutljivi na teksture. Vsak filter pretvori na različen način digitalno sliko v množico atributov. Proizvedena množica atributov ima različno število in tip atributov v odvisnosti od uporabljenega filtra.
Za izdelavo zaključne naloge smo aplicirali tri različne filtre nad podatkovno zbirko in sicer: PHOG (angl. **P**yramid **H**istogram of **O**riented **G**radients) [29], Edge Histogram Filter in JPEG Coefficient Filter.

6.2.1 PHOG Filter

PHOG (angl. **P**yramid **H**istogram of **O**riented **G**radients) [29] filter je uporabljen za opisati prostorsko obliko v slikah, oziroma filter predstavi prostorsko porazdelitev robov v obliki vektorja. PHOG filter je osnovan na dveh konceptih: uporaba piramidne predstavitve [30] in HOG (angl. **H**istogram of **O**riented **G**radients) [31].
Filter deluje tako da najprej izvleče vzorec robnih kontur iz originalne slike. To je prikazano v sliki 15 (b), kjer informacija robnih kontur je opis oblike slike.

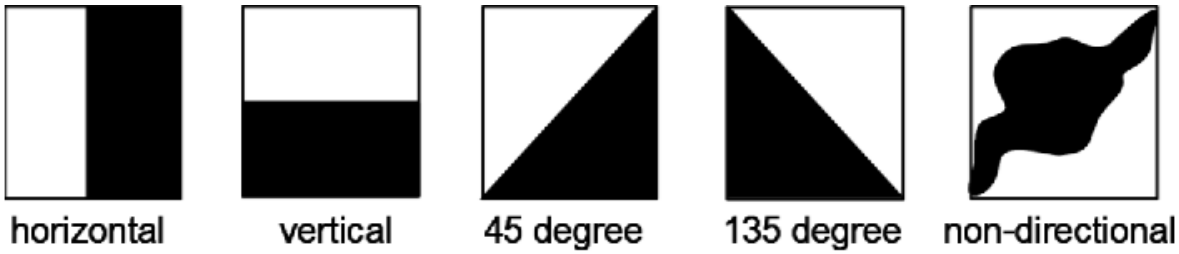


Slika 14: primer delovanja PHOG filter (vir: [29])

Nato slika je razdeljena na več celic pri različnih piramidnih nivojih. Kot prikazano na sliki 15 (a) mreža na piramidnem nivoju l ima 2^l celic. Za vsako mrežo pri vsakem piramidnem nivoju ločljivosti je izračunan HOG (Histogram Usmerjenih Gradientov), kjer vsaka vrednost v histogramu predstavlja število robov, ki imajo isto usmerjenost v nekem kotnem intervalu, slika 15 (c). Končna PHOG komponenta za sliko je združitev vseh HOG vektorjev ob vsakem piramidnem nivoju ločljivosti. Vsak HOG vektor je normaliziran ob upoštevanju vseh piramidnih nivojev. Rezultat je vektor, ki vsebuje vse normalizirane HOG komponente in nam nudi prostorske informacije o sliki [29].

6.2.2 Edge Histogram Filter

Robovi v sliki so pomembna nizko nivojska lastnost uporabljena za vsebinsko analizo slike, robovi lahko opišejo obliko, kot tudi teksturo [32]. Edge Histogram Filter ali EHD (angl. **E**dge **H**istogram **D**escriptor) opisuje pet različnih vrst robov na sliki, in sicer horizontalne, navpične, 45-stopinj diagonalne, 135-stopinj diagonalne in neusmerjene robove.
EHD deluje tako, da slika je najprej razdeljena v 4x4 pod-slik neodvisno od velikosti dane slike. Iz vsake pod-slike se generira histogram robov, ki je sestavljen iz 5 posod, kjer vsaka posoda vsebuje število robov posamezne vrste v pod-sliki. Za identificirati robove je potrebno razdeliti pod-sliko v ne prekrivajoče bloke majhne velikosti. Če blok vsebuje rob potem se poveča število v histogramu za tisto vrsto roba, drugače če v bloku ni prisoten rob se ne poveča števil v histogramu. Ko se je analiziralo vsak blok v pod-sliki je potrebno normalizirati vrednosti v histogramu za število blokov v pod-sliki. Ker originalna slika je bila razdeljena na 4x4 dele in vsaka pod-slika je generirala histogram robov z petimi posodami potem končni EHD je sestavljen iz 80 (16x5) posod.



Slika 15: različni tipi robov pri EHD (vir: [33])

6.2.3 JPEG Coefficient Filter

JPEG Coefficient Filter deluje na podlagi diskretne kosinusne transformacije (DCT). DCT je pogosto uporabljena za kompresijo podatkov in predstavlja osnovo JPEG formata za kompresijo digitalnih slik [34]. Ekstrakcija DCT koeficientov je lahko uporabljena, kot neka vrsta podpisa, ki je uporaben za prepoznavanje izrazov obraza in podobnih nalog. Aplikacija DCT razdeli sliko na več spektralnih pod-pasovih različne pomembnosti oziroma konceptualno vsak DCT koeficient predstavlja različno dimenzijsko značilnost [35]. DCT je konceptualno podobna Diskretni Furjerjevi Transformaciji, saj pretvori signale oziroma neko sliko iz prostorske domene v frekvenčno domeno.
Osnovna enačba za 2D (N x M sliko) DCT je naslednja:

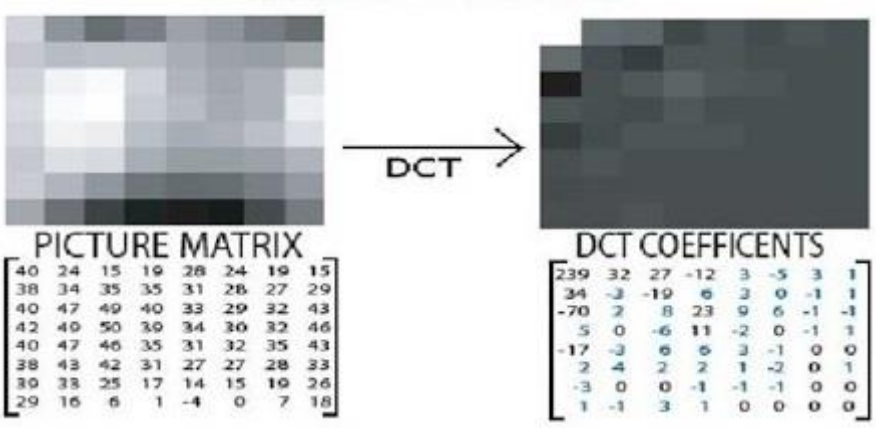
$$F(u,v) = \sqrt{\frac{2}{N}}\sqrt{\frac{2}{M}}\sum_{i=1}^{N-1} A(i) * \cos\left(\frac{u(2i+1)\pi}{2N}\right) * \sum_{j=0}^{M-1} A(j) * \cos\left(\frac{v(2j+1)\pi}{2M}\right) * f(i,j)$$

kjer

$$A(i) = \begin{cases} \frac{1}{\sqrt{2}} \; for \; u = 0 \\ 1 \; drugače \end{cases}$$

$$A(j) = \begin{cases} \frac{1}{\sqrt{2}} \; for \; v = 0 \\ 1 \; drugače \end{cases}$$

- Velikost vhodne slike je N x M
- f(i,j) je intenziteta pixla na poziciji x(i,j)
- F(u,v) je DCT koeficient za pixel x(i,j)
- Pri večini slik največjo energijo signala imajo nizke frekvence ter izražene v zgornjem levem kotu DCT-ja
- Kompresija je dosežena z rezanjem visokih frekvenc, ki nastopajo v spodnjem desnem kotu DCT-ja. Rezanje visokih frekvenc omogoči kompresijo slike z minimalno izgubo. Izhodni vektor DCT koeficientov vsebuje naravna števila od -1024 do 1023 [35].



Slika 16: diskretna kosinusna transformacija 8x8 sivinske slike (vir: [36])

Slika prikazuje aplikacijo DCT na 8x8 sivinsko sliko. Takoj je opaziti da DCT koeficienti v levem zgornjem kotu imajo višje vrednosti od tistih v desnem spodnjem kotu, ker koeficienti v levem zgornjem kotu predstavljajo nizke frekvence, ki imajo večjo energijo signala.
JPEG Coefficient Filter pretvori sliko v DCT komponente in nato tvori Histogram JPEG koeficientov, ki vsebuje značilnosti slike izražene preko DCT transformacije.

6.2.4 Aplikacija filtrov

Za aplicirati filter v Weki je potrebno najprej odpreti Raziskovalec v zavihku Preprocess in nato naložiti ARFF datoteko z podatkovno zbirko. Filter se izbere z gumbom Choose Filter in po izbiri je potrebno definirati kje se nahajajo slike. Filter se požene z gumbom Apply ter je potrebno počakati da Weka obdela podatkovno zbirko in pretvori vsako sliko v množico atributov. Aplikacija filtra zgenerira nove atribute, ki so takoj prikazani v Raziskovalcu ter obstaja možnost za shraniti rezultate v obliki ARFF datoteke oziroma ponovno aplicirati različne filtre ali druga orodja.

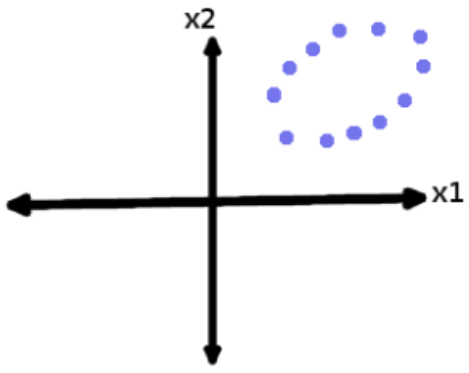
- Aplikacija **PHOG** filtra zgenerira novih 630 atributov. Atributi so numeričnega tipa in lahko zavzamejo vrednosti od 0 do 15
- Aplikacija **Edge Histogram Filter** proizvede množico 80-ih atributov numerične vrednosti, ki lahko zavzamejo vrednosti od 0 do 7.
- Aplikacija **JPEG Coefficient Filter** ustvari množico 192-ih atributov numerične vrednosti, ki lahko zavzamejo vrednosti od 0 do 255.

Združili smo vse množice atributov apliciranih filtrov v skupno ARFF datoteko, ki sedaj vsebuje atribut z imenom slike, razredni atribut ter drugih 902 atributov, ki so rezultat aplikacije filtrov.

6.2.5 Metoda vodilnih komponent

Metoda vodilnih komponent (PCA angl. **P**rincipal **C**omponent **A**nalisys) izkoristi napredna matematična načela za preoblikovati množico linearno povezanih spremenljivk v manjše število spremenljivk imenovane vodilne komponente [37]. PCA izrazi podobnosti in razlike v podatkih ter omogoči identifikacijo vzorcev tudi v podatkih večjih dimenzij, kjer grafična predstavitev podatkov ni mogoča. Ko je PCA metoda identificirala vzorce v podatkih je mogoče stisniti podatke tako da se zmanjša število dimenzij, brez velikih izgub informacije [38]. PCA je široko uporabljena statistična tehnika v področju analize podatkov, prepoznavanju slik in stiskanju slik.

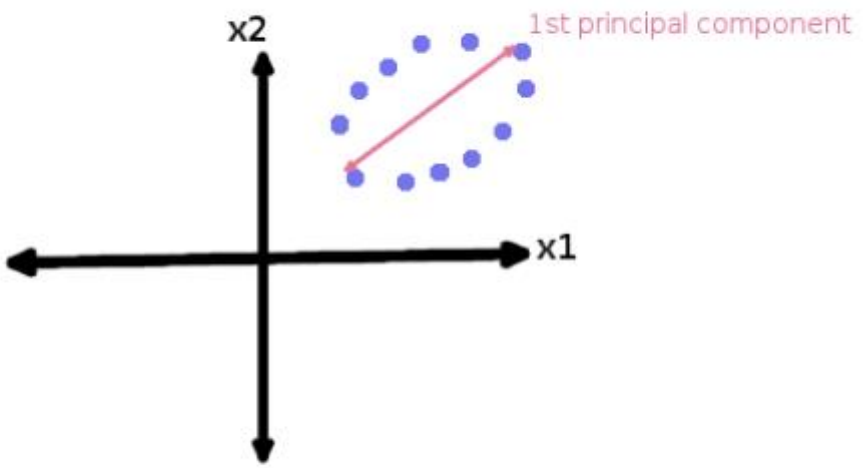
Kako deluje metoda vodilnih komponent?
PCA deluje tako da najde vodilne komponente v podatkih, ter pretvori podatke v novo množico nižje dimenzije v novem koordinatnem sistemu. V novem koordinatnem sistemu prva os predstavlja prvo vodilno komponento, katera vsebuje največjo varianco v podatkih. Za primer vzamemo podatkovno zbirko, ki vsebuje dva atributa X_1 in X_2:



Slika 17: primer podatkovne zbirke z atributi X1 in X2 (vir: [39])

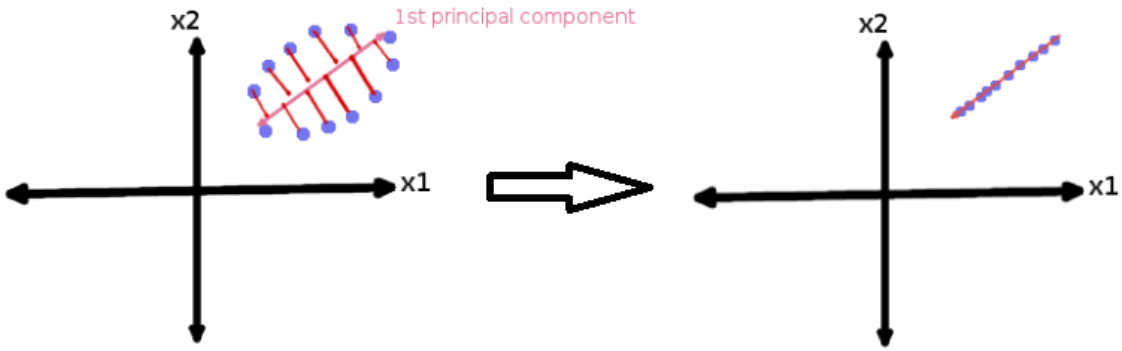
Sedaj želimo identificirati prvo vodilno komponento, ki vsebuje največjo količino variance.

Ker podatki zavzamejo ovalno obliko je enostavno grafično narisati prvo vodilno komponento. Oziroma je potrebno narisati črto, ki prereže oval po dolžini. Ta črta označi komponento, ki vsebuje največ variance.



Slika 18: prva vodilna komponenta (vir: [39])

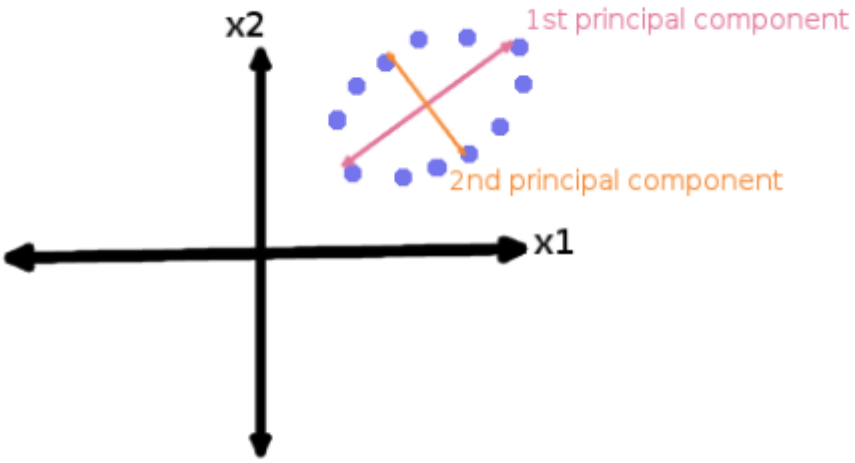
Sedaj želimo projicirati podatke na prvo vodilno komponento tako da reduciramo dvo-dimenzionalno zbirko podatkov v eno-dimenzionalno podatkovno zbirko.



Slika 19: redukcija dimenzije podatkovne zbirke (vir: [39])

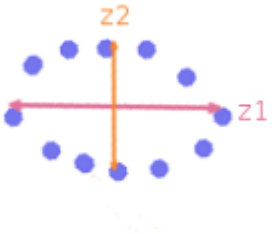
Hitro je opaziti da pri pretvorbi iz dvo-dimenzionalnega prostora v eno-dimenzionalni prostor je prišlo do izgube informacij ampak smo ohranili najvažnejšo os, ki vsebuje informacijo iz obeh atributov X_1 in X_2.

Druga vodilna komponenta:
Druga vodilna komponenta mora biti pravokotna nad prvo vodilno komponento in omogoči da se zazna varianco v podatkih, ki prva vodilna komponenta ni utegnila zajeti.



Slika 20: prikaz druge vodilne komponente (vir: [39])

Za podatkovno zbirko obdelano v primeru lahko obstajajo le dve vodilne komponente. Če bi aplicirali PCA metodo nad to podatkovno zbirko in bi obdržali dve vodilni komponenti potem nebi prišlo do izgube informacij, ker smo enostavno pretvorili dvo-dimenzionalno podatkovno zbirko v novo dvo-dimenzionalno podatkovno zbirko oziroma podatki so le preoblikovani v nov koordinatni sistem definiran od vodilnih komponent.



Slika 21: rezultat PCA metode nad dvo-dimenzionalno podatkovno zbirko (vir: [39])

Koliko vodilnih komponent potrebujemo?
V splošnem podatki sledijo 80/20 pravilu [39] oziroma večina informacije je zajeta le v majhnem številu vodilnih komponent. Najprimerneje je identificirati vodilne komponente, ki opišejo 95% variance v podatkih.
Ni mogoče imeti več vodilnih komponent, kot je atributov v podatkovni zbirki.

6.2.6 Aplikacija metode vodilnih komponent nad podatki

Ker po aplikaciji filtrov smo generirali zelo obsežno podatkovno zbirko, kar 904 atributov je bilo potrebno zmanjšati število atributov in izločiti neuporabne atribute. Za izvesti to operacijo smo uporabili metodo vodilnih komponent, ki reducira dimenzijo podatkovne zbirke brez velikih izgub informacije. PCA metoda je dostopna v Raziskovalcu pod zavihkom Select attributes, kjer je potrebno izbrati metodo vodilnih komponent in algoritem za razvrstitev atributov. Za učinkovito aplikacijo metode je bilo potrebno odstraniti atribut filename, ki vsebuje ime slike, drugače ta atribut bi napovedal razred podatkovnega objekta in metoda vodilnih komponent se nebi izvedla pravilno.
Metoda vodilnih komponent je bila aplicirana z naslednjimi opcijami:

- centerData – false
- doNotCheckCapabilities – false
- maximumAttributeNames – 5
- transformBackToOriginal – false
- varianceCovered – 0.95

PCA varianca je nastavljena na 0.95 oziroma izbrane vodilne komponente bodo vsebovale 95% celotne variance v podatkovni zbirki.
Metoda za razvrščanje atributov pa je Ranker z privzeti parametri.
Aplikacija metode vodilnih komponent je pretvorila 902 atributov v 321 atributov, ki vsebujejo 95% variance podatkovne zbirke. Z redukcijo atributov podatkovne zbirke bo olajšana faza modeliranja oziroma čas gradnje modela se bo opazno skrajšal.

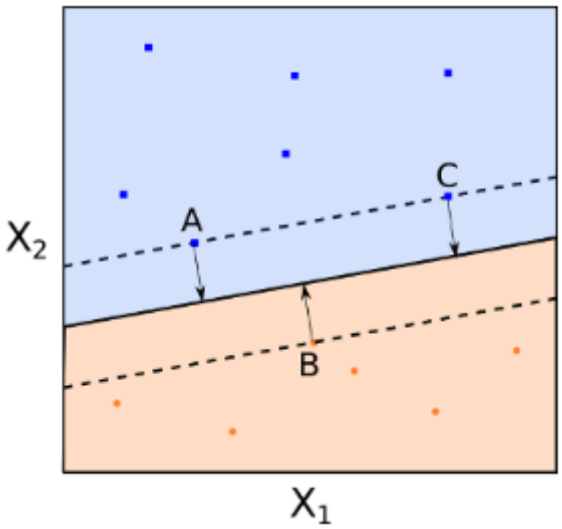
7 MODELIRANJE

7.1 Gradnja modelov

Glavni cilj zaključne naloge je uporabiti napredne metode podatkovnega rudarjenja za analizirati in učinkovito oceniti če na digitalni sliki je prisotna mačka ali pes. Celoten proces podatkovnega rudarjenja sloni na analizi in pripravi podatkov za sledečo gradnjo modelov.
V fazi modeliranja je potrebno izbrati in podrobno preučiti najučinkovitejše metode, ki se bo uporabilo za zgraditi modele.
Za gradnjo modelov si bomo pomagali z Weko, kjer bodo uporabljeni različni klasifikacijski algoritmi. V zaključni nalogi so bili izbrani naslednji klasifikacijski algoritmi: **sekvenčna minimalna optimizacija**, **logistična regresija**, **naključni gozd**, **J48**, **naivni Bayes**.

7.1.1 Sekvenčna minimalna optimizacija

Sekvenčna minimalna optimizacija (angl. **S**equential **M**inimal **O**ptimisation - **SMO**) je algoritem uporabljen za reševanje problema kvadratičnega programiranja, ki se pojavi pri učenju metode podpornih vektorjev. SMO algoritem je izumil John Platt leta 1998 pri Microsoft Research [40], algoritem je široko uporabljen za učenje metode podpornih vektorjev ter je implementiran v knjižnici LIBSVM tool [41].
Metoda podpornih vektorjev (angl. **S**upport **V**ector **M**achines - **SVM**) je algoritem strojnega učenja, ki deluje na podlagi nadzorovanega učenja uporabljen za klasifikacijo ali regresijo. SVM deluje na osnovi iskanja hiperravnine, ki najučinkovitejše razdeli učno množico v dva razreda.



Slika 22: hiperravnina, ki razdeli učno množico v dva razreda (vir: [42])

Na sliki je prikazana hiperravnina, ki razdeli učno množico v dva razreda, točke oziroma vektorji, ki so najbližji hiperravnini se imenujejo **podporni vektorji** in so označeni z črkami A,B,C. Podporni vektorji so podatkovne točke, katerih odstranitev povzroči spremembo v hiperravnini.

Hiperravnina v ***p***-razsežnem prostoru je ravninska podmnožica z razsežnostjo ***p-1*** v zgornjem primeru ko dimenzija prostora ***p = 2*** potem hiperravnina je eno-dimenzijska črta, ki obstaja v večji dvo-dimenzijski ravnini [42].

Če upoštevamo en element našega ***p***-dimenzijskega prostora npr. $\vec{x} = (x_1, \dots, x_p) \in \mathbb{R}^p$, potem lahko enostavno definiramo afino hiperravnino z naslednjo enačbo:

$$b_0 + b_1 x_1 + \cdots + b_p x_p = 0$$

,kjer $\boldsymbol{b_0} \neq 0$ ustvari afino ravnino, to je ravnina, ki ne seka izhodišče. Zgornjo enačbo lahko tako preoblikujemo : $\vec{b} \cdot \vec{x} + b_0 = 0$, ter če nek element $\vec{x} \in \mathbb{R}^p$ zadostuje enačbi potem ta element živi na ***p-1*** dimenzijski hiperravnini. Ta hiperravnina razdeli ***p***-razsežni prostor v dve klasifikacijski regiji, kjer elementi $\vec{x}$ nad hiperravnino zadostujejo:

$$\vec{b} \cdot \vec{x} + b_0 > 0$$

In elementi pod ravnino zadostujejo:

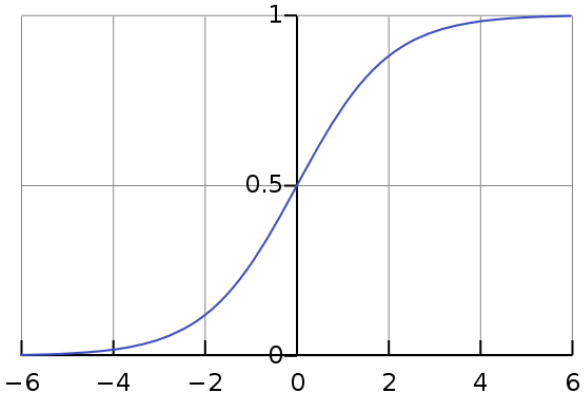
$$\vec{b} \cdot \vec{x} + b_0 < 0$$

Taka enačba omogoči enostavno klasifikacijo elementov $\vec{x}$ oziroma zadostuje izračunati enačbo $\vec{b} \cdot \vec{x} + b_0$, ter predznak rezultata določi v kateri razred spada element $\vec{x}$ [42].
Ker hiperravnina, ki razdeli učno množico ni edina ampak jih obstaja več različno rotiranih ali tranzliranih ravnin, proces iskanja optimalne hiperravnine je zahteven optimizacijski problem kvadratičnega programiranja (angl. **Q**uadratic **P**rogramming – **QP**). **Sekvenčna minimalna optimizacija** deluje na podlagi pristopa devide et impera ter razdeli velik problem kvadratičnega programiranja v množico čim manjših QP problemov. Manjši QP problemi so rešeni analitično brez potrebe po zahtevni QP numerični optimizaciji. Količina zasedenega pomnilnika za izvajati SMO je sorazmerna velikosti učne množice, zato SMO obvlada tudi probleme z velikimi učnimi množicami [40].

7.1.2 Logistična regresija

Logistična regresija je tehnika, ki se je razvila iz področja statistike in je uporabljena tudi v podatkovnem rudarjenju. Tehniko je razvil statistik David Cox leta 1958 [43]. Binarni logistični model je uporabljen za oceniti verjetnost binarnega odziva na podlagi ene ali več neodvisnih spremenljivk. Model preprosto modelira verjetnost izhoda v odvisnosti od danega vhoda in ne izvaja statistične klasifikacije, čeprav je mogoče uporabiti model za izdelavo klasifikatorja, kjer z izbiro mejne vrednosti razvrstimo vhode v dva razreda.
Logistična regresija je imenovana po funkciji uporabljeni v jedru te metode t.i. logistična funkcija [44]. Logistična funkcija ali drugače imenovana sigmoidna funkcija je matematična funkcija oblike črke ***S*** definirana z naslednjo enačbo:

$$S(x) = \frac{1}{(1 + e^{-x})}$$



Slika 23: Logistična funkcija (vir: [45])

Logistična regresija uporablja linearno kombinacijo vrednosti vhoda ***x*** z uporabo koeficientov oziroma uteži ***β*** za napovedati izhodno vrednost ***y.*** Ključna razlika med linearno regresijo in logistično regresijo se nahaja v izhodnih vrednosti, kjer pri linearni regresiji izhodne vrednosti so numeričnega tipa medtem ko pri logistični regresiji izhodne vrednosti so modelirane v obliki binarnih vrednosti 0 ali 1 (true-false , moški-ženska itd.).
Logistična regresija modelira verjetnost da nek vhod ***x*** pripada razredu ***y = 1*** oziroma :

$$P(X) = P(y = 1|X)$$

Opazimo da pri tem napovedujemo verjetnosti, ki je potrebno pretvoriti v binarne vrednosti 0 ali 1 za izvesti napovedovanje. Logistična regresija je linearna metoda, ampak napovedi so pretvorjene z uporabo logistične funkcije, zato ni mogoče upoštevati napovedi, kot linearne kombinacije vhodnih vrednosti.
Logistična funkcija je definirana z enačbo : $S(t) = \frac{1}{(1+e^{-t})}$,
Upoštevamo da ***t*** je linearna funkcija neodvisne spremenljivke ***x*** oblike: $t = \beta_0 + x * \beta_1$
Sedaj logistična funkcija je oblike :

$$P(x) = \frac{1}{(1 + e^{-(\beta_0 + x * \beta_1)})}$$

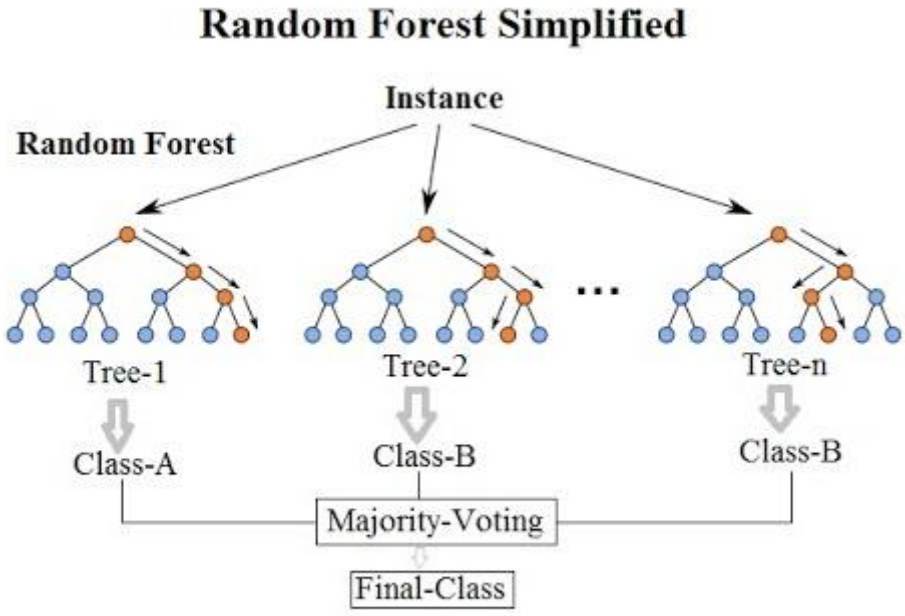
Če apliciramo naravni logaritem ***ln()*** na obe strani, potem lahko odpravimo e^x in tako dobimo:

$$ln\left(\frac{P(x)}{1 - P(x)}\right) = \beta_0 + x * \beta_1$$

Takoj je opaziti da na desni strani imamo ponovno izračun izhoda v linearni obliki in vhod na levi strani je logaritem verjetnosti privzetega razreda. Razmerje na levi strani je posebno razmerje, ki v statistiki se imenuje odds in je razmerje med verjetnostjo da nek dogodek se zgodi in verjetnostjo da ta dogodek se ne zgodi. Naravni logaritem od odds v statistiki se imenuje logit [46].
Učenje modela logistične regresije deluje na podlagi maximum-likelihood estimation [47], ki oceni vrednost koeficientov (beta vrednosti-***β***) iz učenja nad učno množico. Optimalno ocenjeni koeficienti, bodo omogočili da model napove vrednosti zelo blizu 0 za en razred in vrednosti zelo blizu 1 za drugi razred.
Napovedovanje z uporabo logistične regresije je enostavno, ker zadostuje vstaviti vhodne vrednosti (v našem primeru vrednost x) v enačbo logistične regresije, ter izračunati rezultat in postopoma klasificirati izhodno vrednost [44].

7.1.3 Naključni gozd

Naključni gozd (angl. Random forests) je algoritem strojnega učenja, ki deluje na podlagi množice učnih algoritmov za pridobiti boljše rezultate kot posamezni algoritmi [48].
V našem primeru algoritem naključni gozd sestavi množico odločitvenih dreves nad učnimi podatki za izvajati naloge klasifikacije ali regresije. Algoritem naključni gozd je učen preko bagging metode, bagging ali Bootstrap Aggregating je metoda, ki naključno razdeli podatkovno množico na več podmnožic ter zgradi model nad vsako podmnožico in ko se izvaja napovedovanje združi napovedi tako da izvoli razred za katerega je največ glasov.
V našem primeru bagging metoda naključno razdeli učno množico na več podmnožic, ter v vsaki podmnožici zgradi neobrezano odločitveno drevo. Odločitvena drevesa zgrajena v vsaki podmnožici niso navadna drevesa, kjer vsako izbrano vozlišče je optimalni delitelj ampak selekcija vozlišč drevesa deluje tako da se naključno izbere *m* vozlišč med katerimi je izbran najučinkovitejši delitelj [49]. Tako nastane naključni gozd, ki je kolekcija odločitvenih dreves, kjer vsako drevo je nekoliko različno.



Slika 24: Prikaz algoritma naključni gozd (vir: [50])

Napovedovanje v algoritmu naključni gozd deluje z agregacijo rezultatov vseh *n*-dreves prisotnih v gozdu. Oziroma vsak podatkovni objekt je ocenjen od vsakega drevesa v gozdu, tako dobimo *n*-napovedi. Vse napovedi se prešteje ter se glasuje večinski razred, kot napovedani razred podatkovnega objekta [49].

Naključni gozd je napreden algoritem, ki deluje na podlagi odločitvenih dreves ter reši njihove slabosti visoke variance in visoke pristranskosti z uvedbo naključnosti [48].

7.1.4 J48 – Odločitveno drevo

J48 je algoritem implementiran v Weki, ki deluje na principu odločitvenih dreves, znan tudi po imenu C4.5, algoritem je razvil Ross Quinlan leta 1993 [51], za prekoračiti omejitve ID3 algoritma [52].
Odločitveno drevo je eden izmed najbolj uporabljenih pristopov v podatkovnem rudarjenju, deluje na podlagi nadzorovanega učenja in je uporabljeno za klasifikacijo, kot tudi regresijo. Odločitvena drevesa se učijo iz podatkov in sestavijo klasifikator v obliki strukture drevesa z uporabo if-then-else odločitev. Gradnja drevesa se začne iz korena proti listom, kjer koren je atribut, ki najučinkovitejše razdeli podatkovno množico v dve podmnožici. Selekcija korenskega atributa deluje z izračunom Entropije (angl. Entropy) in informacijskega prispevka (angl. Information Gain) [52].
Shannonova entropija je mera informacije, ki jo prenaša verjetnostna porazdelitev ***P***:

$$E(P) = -\sum_{j=1}^{n} p_j * \log(p_j)$$

Informacijski prispevek določi prispevek, ki doprinese test T:

$$infoGain(T) = E(P) - \sum_{j=1}^{n} p_j * \mathrm{E}(p_j)$$

kjer $\boldsymbol{p_j}$ je množica vseh mogočih vrednosti, ki lahko zavzame atribut T. Informacijski prispevek je uporabljen za razvrstiti atribute in graditi odločitveno drevo, kjer ob vsakem vozlišču je prisoten atribut z najvišjim informacijskim prispevkom med vsemi atributi, ki niso bili še upoštevani v poti od korena [52].

7.1.5 Naivni Bayes

Naivni Bayes je klasifikacijska metoda, ki deluje na osnovi Bayesovega izreka [9] z predpostavko, da atributi so med sabo neodvisni. Neodvisnost atributov ni zahtevana ampak metoda deluje z to predpostavko da prisotnost neke lastnosti v razredu ni posledica prisotnosti neke druge lastnosti, zato ta metoda je imenovana naivna. Predpostavka neodvisnosti atributov dramatično zmanjša število parametrov, ki je potrebno obdelati za sestaviti model od $2(2^n - 1)$ na $2n$ [53]. Metoda deluje tako da najprej izračuna pojavitve vsakega razreda za posamezen atribut in izdela t.i. tabelo frekvence. Iz tabele frekvenc izračuna verjetnost pojavitve razreda, verjetnost pojavitve posameznega atributa in pogojno verjetnostjo. Izračunane verjetnosti so uporabljene za klasifikacijo preko Bayesove formule:

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)}$$

,kjer C predstavlja razred in X so vrednosti atributov.

7.2 Testiranje modelov

Modeli so zgrajeni in testirani v programu Weka v raziskovalcu. Gradnja modela deluje tako da najprej se naloži podatkovno zbirko v zavihku Preprocess iz raziskovalca. V našem primeru podatkovna zbirka je bila že pred obdelana z filtri iz vtičnika ImageFilter in metodo vodilnih komponent, kot opisano v šestem odstavku (Priprava podatkov). Ko se je podatkovna zbirka pravilno naložila je potrebno odpreti zavihek Classify, ki vsebuje metode za klasifikacijo. Izbere se metodo z Choose classifier ter se nastavi parametre metode z klikom nad ime metode. V zavihku Classify se nastavi tudi način testiranja in razne opcije za izpis modelov. V zaključni nalogi modeli bodo testirani z uporabo N-kratnega prečnega preverjanja.

7.2.1 N-kratno prečno preverjanje

N-Kratno prečno preverjanje je statistična tehnika uporabljena v strojnem učenju za gradnjo in testiranje modelov. Metoda deluje tako da učna množica je razdeljena na N podmnožic z enakim številom elementov in ob vsakem koraku N-ta podmnožica je uporabljena kot testna množica, ter ostale podmnožice sestavijo trenutno učno množico. Proces je ponovljen N krat, kjer vsaka podmnožica podatkov je uporabljena enkrat kot testna množica [54]. Na tak način model je učen nad celotno učno množico. N-kratno prečno preverjanje je uporabno predvsem, ko podatkovna zbirka na razpolago vsebuje premalo primerov za delitev v učno in testno množico, tako zgradimo in testiramo model nad celotno množico podatkov. V zaključni nalogi bo uporabljeno desetkratno prečno preverjanje oziroma učna množica bo razdeljena na 10 podmnožic in model bo zgrajen ter testiran na vsaki od podmnožic.

8 REZULTATI IN VREDNOTENJE

V osmem poglavju rezultati in vrednotenje bodo opisane nastavitve in parametri uporabljenih metod za gradnjo modelov. Nato bodo predstavljeni rezultati testiranja modelov nad učno množico z uporabo desetkratnega prečnega preverjanja, podana bo natančnost modelov in tabela napačnih klasifikacij (angl. confusion matrix).

8.1 Parametri in nastavitve metod za gradnjo modelov

Program Weka vsebuje veliko metod strojnega učenja, ki so lahko uporabljene v procesu podatkovnega rudarjenja. Vsaka metoda deluje na podlagi različnega algoritma ter nudi možnost spreminjanja parametrov in nastavitev, ki bistveno vplivata na rezultat končnega modela. V naslednjih vrsticah bodo predstavljene nastavitve metod uporabljenih pri gradnji modelov za zagotoviti ponovljivost procesa podatkovnega rudarjenja.

J48 odločitveno drevo
Metoda je bila aplicirana z privzetimi parametri.

Naivni Bayes
Metoda je bila aplicirana z privzetimi parametri.

Naključni gozd
Metoda je bila aplicirana z privzetimi parametri.

Logistična regresija
Metoda je bila aplicirana z privzetimi parametri.

Sekvenčna minimalna optimizacija
Metoda sekvenčne minimalne optimizacije je bila aplicirana z privzetimi parametri le z spremembo privzetega jedra. Uporabili smo jedro Puk namesto privzeto jedro PolyKernel. Puk je univerzalno jedro za metodo sekvenčne minimalne optimizacije, ki deluje na osnovi funkcije Pearson VII, ki je razvil Karl Pearson leta 1895 [55]. Uporabili smo jedro Puk, ker je univerzalno jedro in lahko deluje, kot linearno, polinomsko ali RBF jedro ter v navedeni literaturi [55] je opisano da omogoči doseči boljše rezultate, kot z uporabo ostalih jedrih.

8.2 Primerjava rezultatov napovednih modelov

Modeli so bili zgrajeni z uporabo programa Weka nad učno množico digitalnih slik mačk in psov obdelanih z filtri iz vtičnika ImageFilter in z uporabo metode vodilnih komponent. Postopek aplikacije filtrov in uporabe metode vodilnih komponent je bil opisan v šestem poglavju (Priprava podatkov), kjer so tudi opisani podatki pripravljeni za modeliranje. Za gradnjo klasifikacijskih modelov so bili izbrani naslednji algoritmi: J48, naivni Bayes, naključni gozd, logistična regresija in sekvenčna minimalna optimizacija. Modeli so bili testirani v programu Weka z uporabo desetkratnega prečnega preverjanja. Napovedi zgrajenih modelov bodo predstavljene skupaj z tabelo napačnih klasifikacij.

J48 odločitveno drevo

J48

	število	odstotek (%)
pravilno klasificirane instance	16226	64.904
nepravilno klasificirane instance	8774	35.096

Preglednica 1 : rezultat metode J48

Tabela napačnih klasifikacij

	število pravilnih klasifikacij	število napačnih klasifikacij
instance klasificirane kot mačka	8081	4355
instance klasificirane kot pes	8145	4419

Preglednica 2 : tabela napačnih klasifikacij metode J48

Algoritem J48, ki zgradi odločitveno drevo nad učno množico je dosegel 64.904% natančnosti, kar nakazuje, da ni učinkovita metoda za detekcijo mačk in psov na digitalnih slikah. Iz tabele napačnih klasifikacij je razvidno da imamo skoraj enakomerno porazdelitev napačnih in pravilnih klasifikacij, iz tega sledi da je model generiran od algoritma J48 dobro uravnovešen.

Naivni Bayes

Naivni Bayes

	število	odstotek (%)
pravilno klasificirane instance	16674	66.696
nepravilno klasificirane instance	8326	33.304

Preglednica 3 : rezultati metode naivni Bayes

Tabela napačnih klasifikacij

	število pravilnih klasifikacij	število napačnih klasifikacij
instance klasificirane kot mačka	8490	4316
instance klasificirane kot pes	8184	4010

Preglednica 4 : tabela napačnih klasifikacij metode naivni Bayes

Iz preglednic takoj opazimo da metoda naivni Bayes doseže točnost 66.696% , kar je boljši rezultat od metode J48. Algoritem naivni Bayes je široko uporabljen v področju podatkovnega rudarjenja in strojnega učenja, ker je enostavna ter hitra metoda in doseže dobre rezultate v večini primerov.

Naključni gozd

Naključni gozd

	število	odstotek (%)
pravilno klasificirane instance	18645	74.580
nepravilno klasificirane instance	6355	25.420

Preglednica 5 : rezultati metode naključni gozd

Tabela napačnih klasifikacij

	število pravilnih klasifikacij	število napačnih klasifikacij
instance klasificirane kot mačka	9950	3805
instance klasificirane kot pes	8695	2550

Preglednica 6 :tabela napačnih klasifikacij metode naključni gozd

Metoda naključni gozd je zmožna klasificirati digitalne slike mačk in psov z višjo natančnostjo od prejšnjih metod ter je dosegla 74.580% pravilno klasificiranih slik. Model je dosegel višjo natančnostjo, ker je primernejši za podatkovne zbirke z veliko atributi ter je napredna metoda, ki uvede naključnost v postopku gradnje modela.
Iz tabele napačne klasifikacije je razvidno da 2550 slik mačk je bilo klasificiranih kot pes in 3805 slik psov je bilo klasificiranih kot mačka. Iz tega sledi da zgrajen model ne klasificira enakomerno slike mačk in psov ampak klasificira veliko več slik kot mačka, tudi če na sliki je prisoten pes.

Logistična regresija

Logistična regresija

	število	odstotek (%)
pravilno klasificirane instance	19923	79.692
nepravilno klasificirane instance	5077	20.308

Preglednica 7 : rezultati metode logistične regresije

Tabela napačnih klasifikacij

	število pravilnih klasifikacij	število napačnih klasifikacij
instance klasificirane kot mačka	10004	2581
instance klasificirane kot pes	9919	2496

Preglednica 8 : tabela napačnih klasifikacij logistične regresije

Metoda logistične regresije doseže zelo dober rezultat pri klasifikacij digitalnih slik mačk in psov, kar 79.692% točnosti. Iz tabele napačnih klasifikacij je razvidno da imamo skoraj enakomerno porazdelitev napačnih in pravilnih klasifikacij, iz tega sledi da je model generiran od metode logistične regresije dobro uravnovešen.

Sekvenčna minimalna optimizacija

Sekvenčna minimalna optimizacija		
	število	odstotek (%)
pravilno klasificirane instance	20041	80.164
nepravilno klasificirane instance	4959	19.836

Preglednica 9: rezultati metode sekvenčne minimalne optimizacije

Tabela napačnih klasifikacij		
	število pravilnih klasifikacij	število napačnih klasifikacij
instance klasificirane kot mačka	10432	2891
instance klasificirane kot pes	9608	2068

Preglednica 10: tabela napačnih klasifikacij sekvenčne minimalne optimizacije

Algoritem sekvenčne minimalne optimizacije je dosegel najboljši rezultat med opisanimi metodami in je zmožen klasificirati digitalne slike mačk in psov z 80.164% natančnosti. Tako visoka natančnost modela je bila dosežena zaradi uporabe Puk jedra, ki je izboljšala natančnost modela za 0.34% v nasprotju z privzetim PolyKernel jedrom. Izboljšava natančnosti modela omogoči pravilno klasifikacijo dodatnih 85 slik.
Gradnja modela sekvenčne minimalne optimizacije doprinese najboljše rezultate ampak kompleksnost modela zahteva veliko časa v nasprotju z ostalimi uporabljenimi metodami.

9 ZAKLJUČEK IN NADALJNJE DELO

V zaključni nalogi je bil predstavljen primer podatkovnega rudarjenja za detekcijo mačk in psov v digitalnih slikah. Uporabljeni podatki so del večje podatkovne zbirke imenovane Asirra uporabljene za avtomatiziran Turingov test s pomočjo katerega ločimo računalnike in ljudi med seboj tako imenovan CAPTCHA. Podatkovna zbirka Asirra je bila uporabljena za preverjati če so dejansko ljudje, ki dostopajo do spletnih storitev. Prepoznavanje slik je zelo hitra in enostavna operacija za ljudi ampak je zahtevna naloga za računalnike. Cilj procesa podatkovnega rudarjenja je izdelava modela, ki omogoči klasifikacijo digitalnih slik mačk in psov z visoko natančnostjo.
Proces podatkovnega rudarjenja sledi CRISP-DM metodologiji, ki zagotavlja zanesljivost in ponovljivost procesa. Proces se začne z opisom problema sledi razumevanje podatkov ter faza priprave podatkov v kateri se izvede pretvorba digitalnih slik v množico atributov z uporabo vtičnika ImageFilter. Aplikacija filtrov nad podatki proizvede številno množico atributov, katera je posledično skrčena z uporabo metode vodilnih komponent. Sledi faza modeliranja v kateri se uporabi napredne metode strojnega učenja za zgraditi učinkovite klasifikacijske modele.
Dobre rezultate smo dobili skoraj z vsemi metodami in najbolje so se izkazali naslednji algoritmi: naključni gozd, logistična regresija in sekvenčna minimalna optimizacija. Metoda sekvenčne minimalne optimizacije je dosegla najboljše rezultate in je uspešno klasificirala 80.164% instanc. Tako visoka točnost modela omogoča učinkovito klasifikacijo digitalnih slik mačk in psov ter lahko sklepamo da cilj zaključne naloge je bil uspešno dosežen. Rezultat modela bi lahko dodatno izboljšali z dodatnim postopkom v fazi priprave podatkov v katerem bi odstranili iz učne množice vse digitalne slike, ki ne prikazujejo živali ali prikazujejo ilustrirano žival. Ta postopek ni bil izveden v zaključni nalogi, ker zahteva ročno odstranitev digitalnih slik iz podatkovne množice, ki vsebuje 25.000 instanc.
Nadaljnji razvoj zaključne naloge bi lahko vključeval primer testiranja zgrajenega modela proti sistemu Asirra.

Glede izvedbe zaključne naloge sem zelo zadovoljen, saj sem uspešno izvedel proces podatkovnega rudarjenja po CRISP-DM metodologiji in tako zgradil model, ki je sposoben klasificirati digitalne slike mačk in psov z 80.164% točnosti.

10 LITERATURA IN VIRI

[1] H. M. a. P. S. David Hand, Principles of Data Mining, Massachusetts LondonEngland: The MIT Press, 2001.

[2] J. R. D. J. H. ,. J. S. Jeremy Elson, «Asirra: A CAPTCHA that Exploits Interest-Aligned Manual Image Categorization,» Microsoft Research.

[3] R. M. French, «The Turing Test: the first 50 years,» *Trends in Cognitive Sciences* , vol. 4, n. 3, p. 115, 2000.

[4] J. C. (. R. K. (. T. K. (. T. R. (. C. S. (. a. R. W. (. Pete Chapman (NCR), «CRISP-DM 1.0 , Step-by-step data mining guide,» 2000.

[5] «www.cs.waikato.ac.nz,» [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/. [Dostopano: 07 08 2018].

[6] «The university of WAIKATO Computer Science department,» [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/arff.html. [Consultato il giorno 16 07 2018].

[7] «ImageFilter Weka,» [Online]. Available: https://github.com/mmayo888/ImageFilter. [Dostopano: 16 07 2018].

[8] M. K. J. P. Jiawei Han, Data Mining Concepts and Techniques, 225 Wyman Street, Waltham, MA 02451, USA: ELSEVIER, 2012.

[9] T. Bayes, «An Essay Towards Solving a Problem in the Doctrine of Chances,» 1763.

[10] A. Turing, «ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM,» 1936.

[11] W. S. M. A. W. PITTS, «A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY,» 1943.

[12] R. Li, «History of data mining,» Hacker bits, [Online]. Available: https://hackerbits.com/data/history-of-data-mining/. [Dostopano: 29 07 2018].

[13] A. L. Samuel, «Some Studies in Machine Learning Using the Game of Checkers,» 1959.

[14] M. Learning, Tom M. Mitchell, McGraw-Hill Science/Engineering/Math, March 1, 1997.

[15] B. S. ,. K. S. Mark J. Embrechts, «Introduction to Scientific Data Mining: Direct Kernel Methods & Applications,» 2005.

[16] G. P.-S. a. P. S. Usama Fayyad, «From Data Mining to Knowledge Discovery in Databases,» 1996.

[17] R. Devasthali, «Understanding the concept of simple linear regression,» 2018. [Online]. Available: https://towardsdatascience.com/understanding-the-concept-of-simple-linear-regression-a572087c253. [Dostopano: 30 07 2018].

[18] ZENTUT, «Data Mining Processes,» [Online]. Available: http://www.zentut.com/data-mining/data-mining-processes/. [Dostopano: 09 08 2018].

[19] O. Corporation, «Java,» [Online]. Available: https://java.com. [Dostopano: 28 07 2018].

[20] G. O. System, «GNU-GPL,» 2007. [Online]. Available: https://www.gnu.org/licenses/gpl-3.0.en.html. [Dostopano: 30 07 2018].

[21] E. F. G. H. B. P. P. R. I. H. W. Mark Hall, «The WEKA Data Mining Software: An Update,» 2003.

[22] «Weka Wiki,» [Online]. Available: http://weka.wikispaces.com/ARFF%20%28book%20version%29. [Dostopano: 16 07 2018].

[23] «Advanced Data Mining with Weka (4.6: Application: Image classification),» [Online]. Available: https://www.youtube.com/watch?v=XBSJOkuAtCw&t=111s. [Dostopano: 16 07 2018].

[24] «Advanced Data Mining with Weka,» [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/mooc/advanceddataminingwithweka/slides/Class4-AdvancedDataMiningWithWeka-2016.pdf. [Dostopano: 16 07 2018].

[25] «Kaggle.com,» [Online]. Available: https://www.kaggle.com/. [Dostopano: 17 07 2018].

[26] «Kaggle.com Dogs vs. Cats,» [Online]. Available: https://www.kaggle.com/c/dogs-vs-cats. [Dostopano: 17 07 2018].

[27] «Petfinder.com,» [Online]. Available: https://www.petfinder.com/.

[28] G. K. Wallace, «The JPEG still picture compression standard,» 1991.

[29] L. G. L. J. Yang Bai, «A NOVEL FEATURE EXTRACTION METHOD USING PYRAMID HISTOGRAM OF ORIENTATION GRADIENTS FOR SMILE RECOGNITION,» 2009.

[30] C. S. J. P. Svetlana Lazebnik, «Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories,» in *In IEEE Computer Society Conference on Computer, Vision and Pattern Recognition, vol. 2, pp. 2169-2178,* 2006.

[31] N. Dalal and B. Triggs, «Histograms of oriented gradients for,» in *In IEEE Computer Society Conference on , Computer Vision and Pattern Recognition, vol. 1, pp. 886-893,*, 2005.

[32] C. S. Won, «Feature Extraction and Evaluation Using Edge Histogram Descriptor in MPEG-7,» Seoul, 100-715, South Korea, 2004.

[33] P. P. K. P. A. Vikhar, «Improved CBIR system using Edge Histogram Descriptor (EHD) and Support Vector Machine (SVM),» in *Published 2016 in 2016 International Conference on ICT in Business*, 2016.

[34] A. L. A. a. M. A. E.-N. Abdallah S. Abdallah, «A New Face Detection Technique using 2D DCT and Self Organizing Feature Map,» *World Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering,* vol. 1, n. 3, 2007.

[35] P. P. V. M. M. R. Aman R. Chadha, «Face Recognition Using Discrete Cosine Transform for Global and Local Features,» in *2011 International Conference on Recent Advancements in Electrical, Electronics and Control Engineering*, 2011.

[36] «Discrete cosine transform,» [Online]. Available: https://i.ytimg.com/vi/F5pMHaofd7c/hqdefault.jpg. [Dostopano: 25 07 2018].

[37] M. Richardson, «Principal Component Analysis,» May 2009.

[38] L. I. Smith, «A tutorial on Principal Components Analysis,» February 26, 2002.

[39] L. D. Hamilton, «www.lauradhamilton.com,» [Online]. Available: http://www.lauradhamilton.com/introduction-to-principal-component-analysis-pca. [Dostopano: 25 07 2018].

[40] J. C. Platt, «Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines,» Microsoft Research, 1998.

[41] C.-C. C. a. C.-J. Lin, «LIBSVM: A Library for Support Vector Machines,» Department of Computer Science National Taiwan University, Taipei, Taiwan, 2001.

[42] Q. Team, «www.quantstart.com,» [Online]. Available: https://www.quantstart.com/articles/Support-Vector-Machines-A-Guide-for-Beginners. [Dostopano: 31 07 2018].

[43] D. R. Cox, «The regression analysis of binary sequences (with discussion),» *J Roy Stat Soc B,* vol. 20, pp. 215-242, 1958.

[44] J. Brownlee, «machinelearningmastery.com,» [Online]. Available: https://machinelearningmastery.com/logistic-regression-for-machine-learning/. [Dostopano: 01 08 2018].

[45] «wikipedia.org,» [Online]. Available: https://en.wikipedia.org/wiki/File:Logistic-curve.svg. [Dostopano: 02 08 2018].

[46] K. L. L. ,. G. M. I. CHAO-YING JOANNE PENG, «An Introduction to Logistic Regression Analysis and Reporting,» 2013.

[47] J.-X. P.-T. Fang, «Maximum Likelihood Estimation,» in *Growth Curve Models and Statistical Diagnostics*, pp. pp 77-158.

[48] L. Breiman, «RANDOM FORESTS,» Statistics Department University of California, Berkeley, CA 94720, 2001.

[49] A. L. a. M. Wiener, «Classification and Regression by randomForest,» 2002.

[50] K. Ilan Reinstein, «www.kdnuggets.com,» [Online]. Available: https://www.kdnuggets.com/2017/10/random-forests-explained.html. [Dostopano: 6 08 2018].

[51] J. R. Quinlan, C4.5: Programs for Machine Learning, 1993: Morgan Kaufmann Publishers.

[52] A. M. E. E. Badr HSSINA, «A comparative study of decision tree ID3 and C4.5,» TIAD laboratory, Computer Sciences Department, Faculty of sciences and techniques Sultan Moulay Slimane University, Morocco.

[53] T. M. Mitchell, «GENERATIVE AND DISCRIMINATIVE CLASSIFIERS: NAIVE BAYES AND LOGISTIC REGRESSION,» in *Machine Learning*, 2010.

[54] R. Kohavi, «A Study of Cross-Validation and Bootstrap for Accurancy Estimation and Model Selection,» in *International Joint Conference on Artificial Intelligence*, Standford, 1995.

[55] W. M. L. B. *. B. U¨stu¨ n, «Facilitating the application of Support Vector Regression by using a universal Pearson VII function based kernel,» Elsevier, Nijmegen, 2005.

[56] J. Furnkraz e D. L. N. Gamberger, Foundations of Rule Learning, 2012.