Žilinská Univerzita v Žiline Fakulta
riadenia a informatiky

Diplomová práca

Študijný odbor: **Informačné systémy - Spracovanie dát**

Bc. Oľga Chovancová

Nástroj pre fuzzifikáciu numerických hodnôt

Vedúci: **Ing. Miroslav Kvaššay, PhD.**

Reg.č. 6/2016 Máj 2017

ŽILINSKÁ UNIVERZITA V ŽILINE, FAKULTA RIADENIA A INFORMATIKY.

ZADANIE TÉMY DIPLOMOVEJ PRÁCE.

Študijný program : Informačné systémy

Zameranie: Spracovanie dát

Meno a priezvisko

Oľga Chovancová

Osobné číslo

556217

Názov práce v slovenskom aj anglickom jazyku

Nástroj pre fuzzifikáciu numerických hodnôt

Tool for fuzzification of numerical values

Zadanie úlohy, ciele, pokyny pre vypracovanie (Ak je málo miesta, použite opačnú stranu)

Cieľ diplomovej práce:
Cieľom diplomovej práce je experimentálne porovnať algoritmy, ktoré slúžia pre fuzzifikáciu numerických hodnôt.

Obsah:
1. Oboznámenie sa s problematikou fuzzifikácie (transformácie numerických hodnôt na lingvistické).
2. Rozbor existujúcich algoritmov pre fuzzifikáciu numerických hodnôt.
3. Implementácia vybraných algoritmov fuzzifikácie v jazyku C++.
4. Experimentálne porovnanie implementovaných algoritmov na rôznych výstupných dátach.

Meno a pracovisko vedúceho DP: Ing. Miroslav Kvaššay, PhD., KI, ŽU
Meno a pracovisko tútora DP:

vedúci katedry
(dátum a podpis

Zadanie zaregistrované dňa 31. 10. 2016 pod číslom 6/2016 podpis

Čestné prehlásenie

Prehlasujem, že som diplomovú prácu *Nástoj pre fuzzifikáciu numerických hodnôt* som vypracovala samostatne pod vedením Ing. Miroslava Kvaššaya, PhD.. Uviedla v nej všetky použité literárne a iné odborné zdroje v súlade s právnymi predpismi, vnútornými predpismi Žilinskej univerzity a vnútornými aktmi riadenia Žilinskej univerzity a Fakulty riadenia a informatiky.

V Žiline, dňa 3. mája 2017

Oľga Chovancová

Poďakovanie

Na tomto mieste by som chcela poďakovať vedúcemu diplomovej práce Ing. Miroslavovi Kvaššayovi, PhD., za cenné pripomienky a odborné rady, ktorými prispel k vypracovaniu tejto diplomovej práce. Ďakujem aj priateľom a mojej rodine za ich nekonečnú podporu a trpezlivosť.

V Žiline, dňa 3. mája 2017

Oľga Chovancová

Abstrakt

Chovancová Oľga: *Nástroj pre fuzzifikáciu numerických hodnôt* [Diplomová práca] Žilinská Univerzita v Žiline, Fakulta riadenia a informatiky, Katedra informatiky.
Vedúci: Ing. Miroslav Kvaššay, PhD., KI, ŽU
Stupeň odbornej kvalifikácie: Inžinier Informatiky

Cieľom diplomovej práce je experimentálne porovnať algoritmy, ktoré slúžia pre fuzzifikáciu numerických hodnôt. Oboznámenie s problematikou fuzzifikácie, to je transformácia numerických hodnôt na lingvistické. Analýza existujúcich algoritmov pre fuzzifikáciu numerických hodnôt. Implementácia opísaných algoritmov fuzzifikácie. Experimentálne porovnanie implementovaných algoritmov na rôznych výstupných dátach.

Kľúčové slová: fuzzy, entropia, fuzzifikácia, diskretizácia, lingvistické hodnoty.

Abstract

CHOVANCOVÁ OLGA: *Tool for fuzzification of numerical values* [Diploma thesis]
University of Žilina, Faculty of Management Science and Informatics, Department of Informatics
Tutor: Ing. Miroslav Kvaššay, PhD. KI, ŽU
Qualification level: Masters of Informatics

The aim of the thesis is to compare experimental algorithms that are used for fuzzification numerical values. Introduction to the Fuzzification, which is transformation of numeric values to linguistic values. Analysis of the existing algorithms for fuzzification numerical values. Implementation of selected algorithms fuzzification. Experimental comparison of algorithms implemented on different output data.

Key words: fuzzy, entropy, fuzzification, discretization, linguistic values.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam skratiek

Bayesian Bayesian Discretizer

BRDisc Boolean Reasoning Discretizer

CADD Class-Attribute Dependent Discretizer

CAIM Class-Attribute Interdependence Maximization

EBDA Effective Botton-up Discretizer

EqualFrequency Equal Frequency Discretizer

EqualWidth Equal Width Discretizer

FCM Fuzzy c-means

FEBFC Fuzzy entropy-based fuzzy classifier

ID3 Iterative Dichotomizer 3 Discretizer

MDL-Disc Minimum Description Length Discretizer

MDLP Minimum Description Length Principe

StatDisc Class-driven Statistical Discretizer

WEDA Wrapper Estimation of Distribution Algorithm

Zoznam termínov

Defuzzifikácia je prevod fuzzy množín na ostrú hodnotu.

Funkcia príslušnosti charakteristická funkcia fuzzy množín, ktorá charakterizuje stupeň, s akým daný prvok patrí do príslušnej množiny a to v rozsahu od 0 do 1.

Fuzzifikácia je prevod ostrých vstupných hodnôt do fuzzy množín pomocou funkcií príslušnosti.

Fuzzy logika vychádza z teórie fuzzy množín a zameriava sa na vágnosť, ktorou sa snaží matematicky zachytiť.

Fuzzy množina je množina, ktorej prvok patrí do množiny s istou pravdepodobnosťou a tou je stupeň príslušnosti.

Fuzzy znamená neostrý, matný, mlhavý, neurčitý, vágny.

Lingvistická premenná je premenná, ktorej hodnoty sú výrazy nejakého jazyka.

Termy je množina lingvistických hodnôt.

Univerzum je univerzálna množina, na ktorej sú definované termy.

Vágnosť súvisí s naším vnímaním okolitého sveta spôsobom, ako vidíme jednotlivé objekty a ich vlastnosti,na základe ktorých vytvárame pojmy.

Úvod

V súčasných systémoch pre podporu rozhodovania sú využívané rôzne metódy a algoritmy. Tie by mali brať do úvahy nestochastickú neurčitosť vstupných dát. Toto je spôsobené nedostatočnou presnosťou merania dát. Neurčité fuzzy dáta je vyjadrenie vstupných dát, ktoré berie do úvahy práve neurčitosť dát. Spracovanie neurčitých a viachodnotových dát je vyjadrené pomocou matematicky cez fuzzy a viac-hodnotovú logiku.

Lingvistické premenné sú často používaným spôsobom na vyjadrenie neurčitých dát v konečnej množine. Výsledok formalizácie expertných odhadov sú kvalitatívne veličiny. Každý objekt, proces je opísaný skupinou vlastností, ukazovateľov. Reálne čísla sú často používané v modeloch pre podporu rozhodovania. Použitie reálnych čísiel je zložité na presné meranie hodnôt ukazovateľa. Ďalší problém je v dostupnosti nameraných dát. Meranie presných hodnôt ukazovateľov je spojené s relatívne vysokými nákladmi. V súčastnosti nie sú algoritmy a metódy, ktoré vypočítajú presné hodnoty ukazovateľov. Niekedy reálne hodnoty obsahujú zbytočne podrobnú informáciu, ktorá nemôže byť použitá ako základ na výber rozhodnutia.

Lingvistický prístup je dobrý v tom, že umožňuje formalizovať neurčité fuzzy pojmy pomocou fuzzy množín a premenných a následne ich spracovávať cez teóriu fuzzy logiky.

Lingvistická premenná sa od číselnej premennej líši tým, že jej hodnotami nie sú čísla, ale slová alebo výroky prirodzeného alebo formálneho jazyka. Je zrejmé, že takýto kvalitatívny popis s využitím slov je menej presný ako pomocou čísel. Napriek tomu použitie lingvistickej premennej umožňuje približne opísať zložité javy, ktoré nie je možné opísať pomocou obvyklých kvalitatívnych termínov. [1].

Cieľ práce

Cieľom diplomovej práce je experimentálne porovnať algoritmy, ktoré slúžia pre fuzzifikáciu numerických hodnôt a prípadne ich modifikovať.

Postup práce

1. Oboznámenie sa s problematikou fuzzifikácie (transformácie numerických hodnôt na lingvistické).
2. Rozbor existujúcich algoritmov pre fuzzifikáciu numerických hodnôt.
3. Implementácia vybraných algoritmov fuzzifikácie v jazyku C++.
4. Experimentálne porovnanie implementovaných algoritmov na rôznych výstupných dátach.

Kapitola 1

Analýza súčasného stavu

Fuzzy prístupy možno považovať za odpoveď na požiadavku spracovania neurčitosti, resp. nepresnosti. Táto požiadavka je veľmi rozšírená - v určitej forme sa vyskytuje prakticky v každom reálnom systéme aplikujúcom metódy umelej inteligencie - predovšetkým sa však objavuje tam, kde systémy nejakým spôsobom interferujú s človekom alebo využívajú ľudské znalosti [6] .

V súvislosti s neurčitosťou sa rozlišujú tri základné pojmy:

- **Neurčitosť** - vyplýva z nedostatočnej znalosti faktorov alebo udalosti. O neurčitosti sa hovorí, že je aleatórna, ak pramení z vnútorných vlastností nejakého náhodného javu - t.j. ju principiálne nemožno odstrániť. Ak neurčitosť vyplýva z neznalosti, tak je epistemická.

- **Nepresnosť** - O nepresnosti hovoríme, ak je znalosť faktov a udalosti taká kompletná ako len môže byť, ale spôsob ich vyjadrenia nie je presný alebo jednoznačný.

- **Nekonzistentnosť** - O nekonzistentnosti sa hovorí, ak si znalosti, resp. známe fakty navzájom odporujú[6, 8].

Ľudské znalosti vo väčšine prípadov zahŕňajú neurčitosti, nepresnosť, niekedy môžu byť aj nekonzistentné. Fuzzy prístupy umožňujú určitým spôsobom formalizovať a ďalej spracúvať vágne poznatky. Vágnosť možno považovať za typ nepresnosti. Takýto typ znalostí je ťažké a často aj nemožné vhodne formalizovať konvenčnými metódami.

Fuzzy prístupy predstavujú jednu z možných ciest ako k nim pristupovať, a použiť na formalizáciu neurčitosti [6, 8].

Teória fuzzy množín je zovšeobecnením klasickej teórie množín - fuzzy množiny sú vágne v tom či prvok patrí alebo nepatrí do množiny. Na fuzzy množinách možno vykonávať určité operácie čiastočne analogické s tými, ktoré sú v klasickej teórie množín. Fuzzy logika predstavuje prístup, ktorý zovšeobecňuje konvenčnú logiku a produkčné pravidlá zavedením tzv. lingvistických premenných a lingvistických pravidiel. Fuzzy logika umožňuje formulovať vágne pravidlá. Fuzzy aritmetika rozširuje princípy klasickej aritmetiky na vágne - fuzzy - čísla [6].

1.1 Teória fuzzy množín

Nech $X = \{x\}$ je množina prvkov x.

Fuzzy množina

Definícia 1.1.1. *Fuzzy množina $A \subset X$ je predstavovaná množinou dvojíc $\{(x, \mu_A(x))\}$, kde $x \in X$ a $\mu_A : X \longrightarrow \langle 0, 1\rangle$ je funkcia príslušnosti, ktorá predstavuje subjektívnu mieru príslušnosti elementu x k množine A. Veličina $\mu_A(x)$ nadobúda hodnoty od nuly, ktorá označuje absolútnu príslušnosť po hodnotu jedna, ktorá hovorí o absolútnej príslušnosti elementu x do fuzzy množiny A [1, 2].*

Ak je fuzzy množina A definovaná na konečnej univerzálnej množine $X = \{x_1, x_2, \ldots, x_i, \ldots, x_n\}$, potom je vhodné označiť ju nasledovne

$$A = \{(x_1, \mu_A(x_1)), (x_2, \mu_A(x_2)), ..., (x_i, \mu_A(x_i)), ..., (x_n, \mu_A(x_n))\},$$

kde $(x_i, \mu_A(x_i))$ - je dvojica tvorená elementov x_i a jeho funkciou príslušnosti, nazývaná singleton [1].

Fuzzy premenná

Definícia 1.1.2. *Fuzzy premenná je definovaná trojicou (α, X, A), kde α - je meno fuzzy premennej, $X = \{x\}$ - je množina, tvoriaca definičný obor premennej x, A - je*

fuzzy podmnožina fuzzy množiny X, pre každý prvok ktorej je definovaná funkcia $\mu_A(x)$, udávajúca stupeň príslušnosti daného elementu x do množiny A [1].

Lingvistická premenná

Lingvistická, resp. jazyková premenná je zvláštnym typom premennej, ktorá sa od numerických premenných odlišuje tým, že jej hodnoty - tzv. lingvistické hodnoty - nie sú čísla, ale slovné výrazy. Pritom každej lingvistickej hodnote je priradený význam, t. j. určitá fuzzy množina definovaná na spoločnom univerze [6].

Definícia 1.1.3. *Lingvistická premenná je definovaná päticou (β, T, X, G, M), kde β - je meno lingvistickej premennej; T - je množina jej hodnôt, z ktorých každá je fuzzy premennou na množine X; G - je syntaktické pravidlo pre tvorbu nových mien hodnôt lingvistickej premennej β; M - je sémantická procedúra, umožňujúca transformovať novú hodnotu premennej β, určenú procedúrou G, na fuzzy premennú, t.j. vytvoriť zodpovedajúcu fuzzy množinu [1, 3, 4, 5].*

Charakteristická funkcia

V klasickej teórii množín prvok môže do množiny buď patriť alebo nepatriť. Pre klasické množiny možno definovať tzv. charakteristickú funkciu.

Definícia 1.1.4. *Charakteristická funkcia klasickej množiny S je priradenie typu [6]*

$$\mu_S : U \longrightarrow \{0, 1\} \tag{1.1}$$

Priradenie hodnoty 0 - nepatrí, alebo hodnoty 1 - patrí - ku každému prvku $x \in U$, pričom definičný obor charakteristickej funkcie U sa nazýva univerzum. Univerzum je množina všetkých hodnôt, o ktorých rozhodujeme či do danej množiny patria, alebo nepatria. Platí $S \subseteq U$ [6].

Charakteristickú funkciu klasickej množiny možno definovať nasledovne [6, 9]

$$\mu_S(X) = \begin{cases} 1 & x \in S, \\ 0 & x \notin S. \end{cases} \tag{1.2}$$

V teórii fuzzy množín sa zavádza rozšírenie tohto konceptu - prvok môže do množiny patriť aj čiastočne: viac alebo menej. Vágnosť je teda v otázke príslušnosti prvku ku množine [6] .

Stupeň príslušnosti a funkcia príslušnosti

Mieru do akej prvok patri do fuzzy množiny sa vyjadruje stupňom príslušnosti. Nech A je fuzzy množina. Stupeň príslušnosti prvku x ku množine A označujeme $\mu_A(x)$. Hovoríme tiež, že $\mu_A(x)$ je funkcia príslušnosti fuzzy množiny A [6, 9] .

Funkcia príslušnosti je priradenie

$$\mu_A : U \longrightarrow \langle 0, 1 \rangle \tag{1.3}$$

Obor hodnôt je v tomto prípade

$$\mu_A(X) : U \in \langle 0, 1 \rangle \tag{1.4}$$

Pritom sa rozlišujú nasledujúce prípady:

- ak $\mu_A(x) = 0$, hovoríme, že prvok do množiny A nepatrí,
- ak $\mu_A(x) = 1$, hovoríme, že prvok do množiny A patrí,
- ak $\mu_A(x) \in (0, 1)$, hovoríme, že prvok patrí do množiny A čiastočne, so stupňom príslušnosti ak $\mu_A(x)$.

Aj v tomto prípade sa dá použiť značenie $A \subseteq U$, čím sa rozumie, že množina A je definovaná na univerze U. [6]

Definícia 1.1.5. *Funkcia príslušnosti $\mu_A(x)$ kvantitatívne určuje príslušnosť prvkov základnej množiny uvažovaného priestoru $x \in X$ k fuzzy množine A. Hodnota A tejto funkcie značí, že prvok nepatrí do fuzzy množiny. Hodnota 1 opisuje úplne patriaci prvok. Hodnoty medzi 0 a 1 charakterizujú neurčito zaradené prvky [1, 3, 4, 5].*

Spojité a diskrétne fuzzy množiny

Fuzzy množiny možno rozdeliť podľa spojitosti na spojité a diskrétne. V prípade spojitých fuzzy množín je univerzum spojité. To je aj funkcia príslušnosti je spojitá. Naopak v prípade diskrétnych fuzzy množín sú univerzum aj funkcia príslušnosti diskrétne. Obor funkcie príslušnosti je spojitý v oboch prípadoch. [6]

Spôsoby zápisu fuzzy množín

Fuzzy množiny možno zapísať buď diskrétne alebo spojito.

V prípade, že ide o diskrétnu fuzzy množinu, možno použiť nasledujúci zápis [7, 6]

$$A = \{\mu_A(x_1)/x_1, \mu_A(x_2)/x_2, ..., \mu_A(x_n)/x_n\}, \tag{1.5}$$

kde n je počet prvkov, $x_i in U : \forall_i = 1, 2, ..., n$ sú prvky univerza a $\mu_A(x_i)$sú ich stupne príslušnosti.

Ďalšia konvencia zápisu diskrétnych fuzzy množín je [6]

$$A = \{\mu_A(x_1)/x_1 + \mu_A(x_2)/x_2, ... + \mu_A(x_n)/x_n\}, \tag{1.6}$$

$$A = \{(x_1; \mu_A(x_1)), (x_2; \mu_A(x_2)), ..., (x_n; \mu_A(x_n))\}, \tag{1.7}$$

$$A = \sum_{i=1} \mu_A(x_i)/x_i. \tag{1.8}$$

Spojité fuzzy množiny možno reprezentovať výrazom v tvare [7]

$$A = \int_U \mu_A(x)/x dx. \tag{1.9}$$

Singleton

Špeciálnym typom fuzzy množiny je tzv. singleton. Ide o taký typ fuzzy množiny, pre ktorý iba jeden bod univerza má stupeň príslušnosti väčší ako 0. Nech teda A je fuzzy

množina definovaná na univerze U. Potom fuzzy množinu A považujeme za singleton, ak existuje bod $x_0 \in U$ taký, že platí [6]

$$\mu_S(X) = \begin{cases} b & x = x_0 \\ 0 & inak \end{cases} \tag{1.10}$$

$$b \in (0, 1\rangle$$

Singleton možno zapísať obdobným spôsobom ako diskrétnu fuzzy množinu a to je [6]

$$A = \{b/x_0\}, \tag{1.11}$$

$$A = \{(x_0; b)\} \tag{1.12}$$

Normalizácia

Normalizácia znamená prevod vstupných ostrých hodnôt z technického procesu, ktorými sú fyzikálne hodnoty nameraných, či zadaných hodnôt, ktoré sa prevedú na normalizovanú množinu - univerzum. Najčastejšie sú všetky univerzá v rozsahu $[-1; 1]$ alebo $[0, 1]$. Takéto znormalizované dáta vstupujú do fuzzifikácie. Pri fuzzifikácii sa každej ostrej nameranej hodnote z normalizovaného univerza priradí stupeň príslušnosti do jednej, alebo viac fuzzy množín, ktoré zodpovedajú významu základných termov použitých v pravidlách. Normalizované univerzum sa pokryje príslušnými fuzzy množinami. Prevedú sa ostré dáta na fuzzy dáta. [17]

Medzi najbežnejšie používané funkcie príslušenstva patria:

- trojuholníková,
- lichobežníková,
- Gausova,
- zvonová,
- sigmoidalná.

1.2 Transformácia číselných hodnôt na lingvistické premenné

Transformácia číselných premenných na lingvistické predpokladá vzťah medzi kombináciou ich hodnôt a stupnicou s konečným počtom m intervalov sa nazýva diskretizácia. Pritom sa každý interval asociuje s hodnotou lingvistickej premennej (termom). Proces diskretizácie predpokladá transformáciu kvantitatívnych dát na kvalitatívne [1, 11].

Vo všeobecnom prípade diskretizácia zmenšuje objem dát a nevedie k zníženiu klasifikačnej presnosti a spoľahlivosti hodnôt využívaných v systémoch pre podporu rozhodovania. Naopak, takéto diskrétne hodnoty adekvátne ohodnocujú ukazovatele a sú stabilnejšie vzhľadom na zmeny a metodiky merania[1, 12].

Úloha diskretizácie sa definuje nasledovne.

Definícia 1.2.1. *Majme množinu pozostávajúcu z N príkladov. Každý príklad obsahuje množinu vstupných číselných premenných. Nech niektorá vstupná číselná premenná X v týchto príkladoch nadobúda hodnoty v rozsahu od x_min po x_max. Potom diskretizáciou tejto spojitej premennej sa nazýva proces rozkladu jej hodnôt na m diskrétnych intervalov*

$$D = \{\langle d_0, d_1\rangle, \langle d_1, d_2\rangle, ..., \langle d_{m-1}, d_m\rangle\}, \tag{1.13}$$

kde d_0 je minimálna hodnota x_min tejto premennej, d_m je maximálna hodnota x_max premennej a platí $d_i < d_{i+1}$, pre i = 0, 1, ... m-1. $P = \{d_1, d_2, ..., d_{m-1}\}$ je množina bodov rezu premennej X [1, 18].

1.3 Metódy diskretizácie

Získanie optimálnej diskretizácie prestavuje NP-zložitú úlohu. Existencia obrovského množstva metód diskretizácie je vysvetliteľná rozmanitou povahovou vstupných dát a požiadaviek na ich spracovanie. Výber použitej metódy diskretizácie určuje úspešnosť ich budúceho spracovania. Na výber metódy má vplyv množstvo parametrov. [1, 24]

1. **Počet intervalov rozkladu.** Príliš malý počet intervalov spôsobuje nepresnosti a chyby vo vstupných dátach a vedie k hrubému vyjadreniu výsledku. Na druhej

strane rozklad vstupných dát na príliš veľký počet intervalov spôsobuje prílišnú detailizáciu a následne vedie k tomu, že spracovanie takýchto dát je pomalé a neefektívne [1, 11].

2. **Nesúlad s výsledkom.** Nesúlad s výsledkom sa objasňuje vznikom neočakávaných chýb v procese diskretizácie. Tieto chyby súvisia s nepresnosťami diskretizácie vstupných dát a vo vzťahu medzi týmito diskretizovanými hodnotami a hodnotami výstupného atribútu [1].

3. **Presnosť rozkladu.** Predpokladá, že úspešný algoritmus diskretizácie, skonštruovaný na základe výberu tréningových dát, pracuje bez podstatného zníženia kvality aj na všetkých nasledujúcich dátach [1].

4. **Časové ohraničenia.** V prípade statických procesov, kedy požiadavka diskretizácie výučbovej množiny vzniká iba raz, nie je čas výpočtu dôležitým parametrom. Avšak pri dynamických procesoch, keď sa etapa diskretizácie výučbovej množiny opakuje mnohokrát, je čas spracovania kritickým parametrom [1].

Proces transformácie reálnych hodnôt na konečný počet intervalov a reprezentovanie každého intervalu s diskrétnou hodnotou predstavuje proces, ktorý je už dosť dobre preskúmaný. Vykonaná analýza odhalila rad základných kritérií klasifikácie existujúcich metód diskretizácie. Podrobnejšie je prehľad a analýza existujúcich metód diskretizácie uvedená v prácach [12, 18, 19, 20, 21, 22, 23] .

Kritérium 1. - Statické a dynamické metódy diskretizácie

Kritérium uvažuje klasifikáciu metód na základe opakovateľnosti procesu diskretizácie. Statické metódy sa realizujú v tvare predbežnej samostatnej etapy spracovania vstupných dát a nezávisia od spôsobu následného využitia diskretizovaných hodnôt. Dynamická diskretizácia je obvykle zabudovaná do mechanizmu inteligentného spracovania dát a využíva sa napríklad pri konštrukcii rôznych klasifikátorov [1].

Kritérium 2. - Globálne a lokálne metódy diskretizácie

Kritérium berie do úvahy úplnosť množiny a dostupnosť vstupných hodnôt, použitých v procese diskretizácie. Globálne metódy diskretizácie spracovávajú vstupnú množinu hodnôt číselného atribútu počas etapy predbežného spracovania. Lokálne metódy predpokladajú diskretizáciu dát súčasne s inými metódami ich spracovania. K tomuto typu patria dynamické metódy diskretizácie, ktoré zisťujú body rezu v rámci vnútorných operácií algoritmov spracovania dát a nemajú prístup k úplnej množine vstupných dát [1].

Kritérium 3. - Metódy jednorozmernej, mnohorozmernej diskretizácie

Kritérium slúži na klasifikáciu metód diskretizácie na základe počtu súčasne spracovaných atribútov. V jednorozmernej diskretizácii sa každý číselný atribút transformuje na lingvisticky nezávisle od hodnôt iných atribútov. Metódy mnohorozmernej diskretizácie predpokladajú súčasné spracovanie všetkých číselných atribútov. Vo výsledku rozklad hodnôt číselných atribútov na intervaly vykonáva s prihliadnutím na množný vzájomný vplyv vstupných atribútov jedného na druhý [1].

Kritérium 4. - Diskretizácia bez učitela, s učiteľom

Kritérium rozlišuje neriadenú diskretizáciu bez učiteľa alebo riadenú diskretizáciu s učiteľom. Diskretizácia bez učiteľa určuje body rezu iba na základe analýzy vstupných atribútov. Hodnoty výstupného atribútu sa pritom neberú do úvahy. Diskretizácia s učiteľom pri rozklade vstupného atribútu na niekoľko intervalov navyše zisťuje vzťah medzi hodnotami vstupných atribútov a im zodpovedajúcimi hodnotami výstupných atribútov. Výsledkom použitia diskretizácie s učiteľom ej možnosť takého rozdelenia vstupného atribútu, kedy rôznym intervalom zodpovedajú rôzne hodnoty výstupného atribútu. Využitie diskretizácie s učiteľom umožňuje automaticky určovať najlepší počet intervalov a body rozdelenia pre každý atribút s perspektívou realizácie budúcej klasifikácie alebo klusterizácie výstupného atribútu [1].

Kritérium 5. - Priame a inkrementálne metódy diskretizácie

Kritérium predpokladá klasifikáciu podľa spôsobu získania intervalov. Metódy priamej diskretizácie naraz rozdelia vstupnú množinu na m intervalov. Využitie týchto metód na začiatku vyžaduje určenie veličiny m. V každom kroku spracovania vyberajú tieto metódy niekoľko bodov rezu. Na druhej strane metódy postupnej diskretizácie predpokladajú hľadanie najlepšieho kandidáta na bodu rezu [1].

Kritérium 6. - Neparametrické a parametrické metódy diskretizácie

Kritérium predstavuje variant predchádzajúceho kritéria a určuje spôsob získania počtu intervalov pre každý atribút.Neparametrické metódy určujú najlepší počet intervalov pre každý atribút. Parametrické metódy predpokladajú, že počet intervalov je už apriórne zadaný [1].

Kritérium 7. - Metódy zlučovacej a rozdeľovacej diskretizácie

Kritérium klasifikuje metódy diskretizácie v závislosti od spôsobu spracovania vstupných hodnôt - zlučovanie alebo rozdeľovanie. Zlučovacia (merging) diskretizácia predpokladá postupné pridávanie bodov rezu a jeho výsledkom je rozklad na menšie intervaly. Rozdeľovacia (splitting) diskretizácia spočíva v odstraňovaní bodov rezu, čo vedie k postupnému zlučovaniu skupiny susedných intervalov do väčšieho intervalu. Opakovanie procesov rozdeľovania alebo zlučovania sa riadi ukončovacím kritériom. Ako primitívne kritérium ukončenia vystupuje napríklad apriórne zadaný počet intervalov.Ako zložité kritérium sa používa napríklad minimálna chyba klasifikácie. Toto kritérium je prípustné iba pre metódy inkrementálnej diskretizácie. Sú známe aj metódy diskretizácie, ktorých činnosť je založená na zlučovaní alebo rozdeľovaní niekoľkých intervalov. Hybridné metódy počas činnosti striedajú spájanie a rozdeľovanie [1].

Kritérium 8. - Metódy diskretizácie podľa použitého parametra

Kritérium klasifikuje metódy diskretizácie podľa použitého parametra porovnania rozličných variantov rozdelenia. Metóda založená na informačných ukazovateľoch často vy-

užíva pojem entropie a iné ukazovatele, založené na pojmoch teórie informácie. Štatistické metódy berú do úvahy mieru závislosti (korelácie) medzi atribútmi. Metódy, ktoré berú do úvahy frekvenčné charakteristiky, patria k najjednoduchším metódam diskretizácie. V týchto metódach sa každý interval vopred určuje, určením apriórne zadaného počtu hodnôt. Kolekcia metód využívajúcich chybu klasifikácie, presnosti klasifikácie [1].

Kritérium 9. - Metódy diskretizácie vytvárajúce nepretínajúce a pretínajúce sa intervaly

Klasifikácia na základe vlastností získaných intervaloch. K metódam diskretizácie vytvárajúce nepretínajúce sa intervaly patria doteraz všetky vymenované metódy diskretizácie. Tieto metódy vytvárajú v procese diskretizácie m diskrétnych intervalov

$$\langle d_0, d_1), \langle d_1, d_2), ..., \langle d_{m-1}, d_m \rangle,$$

pre ktoré platia nerovnosti

$$d_i < d_{i+1},$$

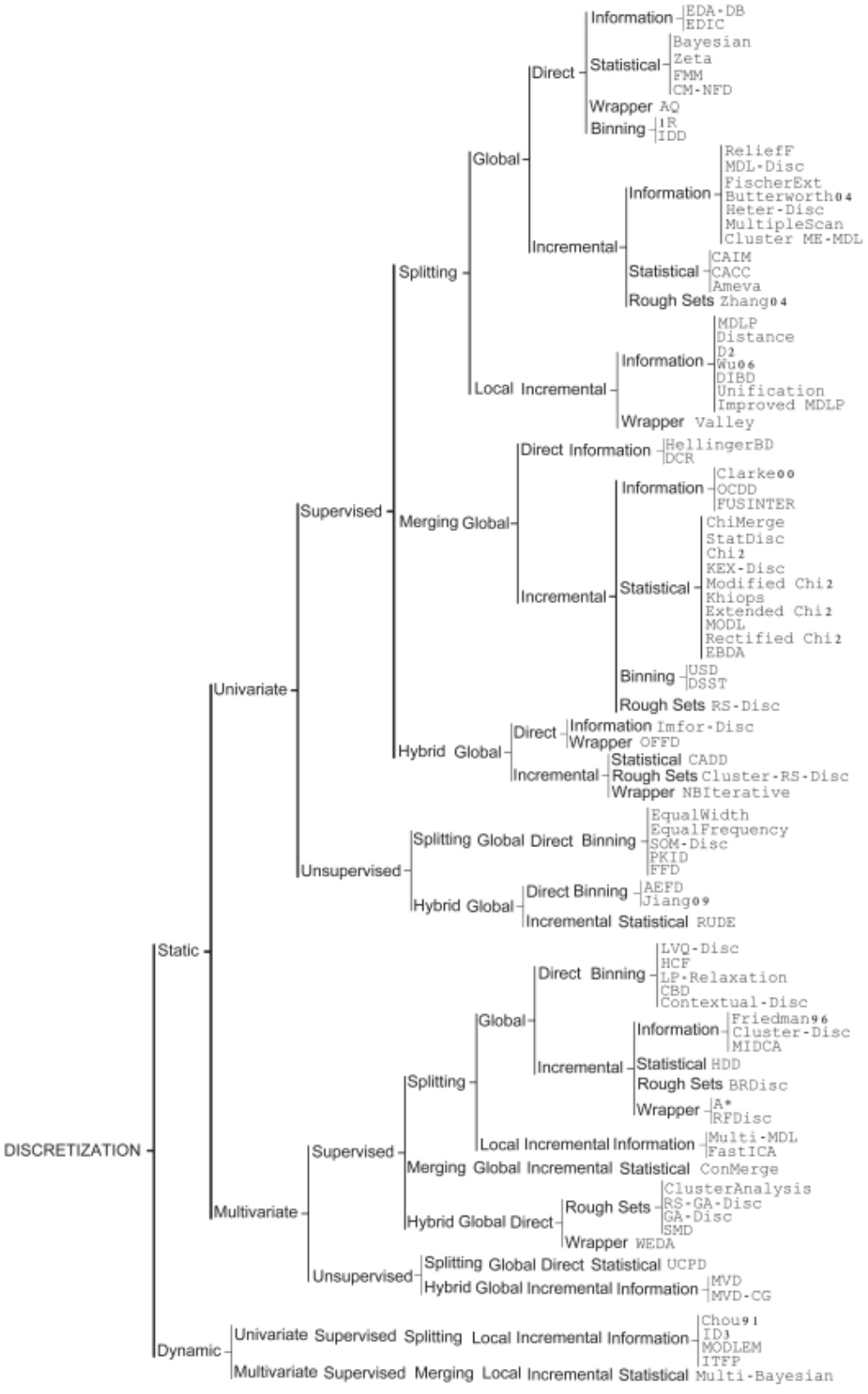
pre i = 0, 1, ... m-1. Pre metódy diskretizácie, ktoré pretínajú intervaly nemusí platiť nerovnosť. K takýmto metódam patrí fuzzy diskretizácia[1].

1.3.1 Príklady algoritmov metód diskretizácie

Obrázok 1.1 zobrazuje kategorizáciu hierarchie metód diskretizácie. V tabuľke 1.1 sú vypísané príklady algoritmov jednotlivých kritérií diskretizácie.[18]

1.4 Meranie Entropie

Entropia je meraná množstvom neistoty výsledku náhodného experimentu, alebo equivalene, meraním informácií, keď sa pozoruje výsledok. Tento koncept bol zadefinovaný rôznymi spôsobmi [13, 14] a zovšeobecnený v rozličných aplikovaných oblastiach, ako napríklad teória komunikácie, matematiky, štatistickej termodynamike a ekonómii [15, 16]. Z pomedzi týchto rozličných definícií, Shannon prispel k najširšej a naj fundamentálnejšej definícii entropie v informačnej teórii.



Obr. 1.1: Taxonómia metód diskretizácie[18]

Kritérium diskretizácie	**Príklady algoritmov**
Statická	ChiMerge, MDLP, Chi2, RFDisc, UnDisc, CDisc, ME-MDL, ImformDisc, ImpMDLP
Dynamická	ID3 a C45, MultiBayesian, MODLEM, ITFP
Globálna	EqualWidth a EqualFrequency, ChiMerge, Chi2, CAIM, RFDisc, UnDisc, ME-MDL, EBDA, ImforDisc
Lokálna	MDLP, ID3 a C45, ImpMDLP
Jednorozmerná	EqualWidth a EqualFrequency, MDLP, UnDisc, EBDA, ME-MDL, ImformDisc
Mnohorozmerná	MVD, Multi-MDL, UCPD, MIDCA, WEDA, RFDisc, SMD, CBD
Neriadená (bez učiteľa)	EqualWidth a EqualFrequency, MVD, UnDisc, FFD
Riadená (s učiteľom)	ChiMerge, MDLP, ID3 a C45, Chi2, WEDA, RFDisc, CDisc, EBDA, ME-MDL, ImforDisc, ImpMDLP
Priama	UCPD, WEDA
Inkremetálna (postupná)	ChiMerge, ID3 a CD45, Chi2, MVD, RFDisc, EBDA, ImpMDLP
Neparametrická	MDLP, USD, CAIM
Parametrická	ChiMerge, CADD
Zlučovacia	ChiMerge, EBDA
Rozdeľovacia	MDLP, RFDisc, CDisc, ME-MDL, ImpMDLP
Súčasná	HBD, IDD
Hybridná	CADD, WEDA, ImforDisc
Informačných ukazovateľov	ID3 a C45, MDLP, DbEr, ME-MDL, ImforDisc, ImpMDLP, GINI
Štatistických ukazovateľov	ChiMerge, Zeta, Chi2, EBDA, MODL
Frekvenčných charakteristík	EqualWidth a EqualFrequency, UnDisc
Presnosti klasifikácie	Valley, NBIterative, RFDisc

Tabuľka 1.1: Tabuľka príkladov algoritmov jednotlivých metód diskretizácie.

Shannonova Entropia

Za entropiu možno považovať meranie neistoty náhodnej premennej X.

Definícia 1.4.1. *Nech X je náhodná spočítateľná premenná s konečnou N-znakovou abecedou danou $\{x_0, x_1, \ldots, x_{N-1}\}$. Ak výsledok x_j sa vyskytuje s pravdepodobnosťou $p(x_j)$, tak potom množstvo informácie spojené so známym výskytom výstupu x_j je definované ako:*

$$I(x_j) = -\log_z p(x_j).$$

To znamená, že pre diskrétne zdroje, informácie získané výberom symbolu x_j sú $\lceil \log_z p(x_j) \rceil$ bitové. V priemere, symbol x_j bude vybratý $n * p(x_j)$-krát z celkového počtu N výberov, takže priemerné množstvo informácie získanej zo zdrojových výsledkov je:

$$-n.p(x_0)\log_2 p(x_0) - n.p(x_1)\log_2 p(x_1) - \ldots - n.p(x_{N-1})\log_2 p(x_{N-1}).$$

Podelením výrazu číslom n sa získa priemerné množstvo informácie na symbol výskytu zdroja. To je známe ako priemerná informácia, neistota, alebo entropia definovaná nasledovne.

Definícia 1.4.2. *Entropia $H(X)$ náhodnej diskrétnej premennej X je definovaná ako*

$$H(x) = -\sum_{j=0}^{N-1} p(x_j)\log_2 p(x_j)$$

alebo

$$H(x) = -\sum_{j=0}^{N-1} p_j \log_2 p_j$$

.

Entropia je funkcia distribúcie X. Nezáleží na skutočných hodnotách náhodnej premennej X, ale iba na pravdepodobnostiach. Preto entropiu možno zapísať ako $H(p)$.

Luca-Termini Axiómy

Kosko [13] navrhuje, aby meranie dobre definovanej fuzzy entropie musí spĺňať štyri Luca-Termini axiómy. Patria medzi ne nasledujúce:

- $E(A) = 0 \iff A \in 2^X$, kde A nie je fuzzy množina a 2^X je množina všetkých podmnožín množiny A.
- $E(\tilde{A}) = 1 \iff m_A(x_i) = 0.5$ pre všetky i, kde $m_{\tilde{A}}(x_i)$ znamená stupeň príslušnosti x_i v fuzzy množine $\tilde{A}$.
- $E(\tilde{A}) \leq E(\tilde{B})$, ak $\tilde{A}$ je menej fuzzy než $\tilde{B}$, napr. ak $m_{\tilde{A}}(x) \leq m_{\tilde{B}}(x)$, keď $m_B(x) \leq 0.5$ a $m_{\tilde{A}}(x) \geq m_{\tilde{B}}(x)$, keď $m_B(x) \geq 0.5$, kde $\tilde{A}$ aj $\tilde{B}$ sú fuzzy množiny.
- $E(A) = E(A^C)$.

Fuzzy entropia na intervale pre každú vlastnosť v rozmere

Definícia 1.4.3. *Definícia fuzzy entropie na základe Shannonovej entropie.*

1) Nech $X = r_1, r_2, \ldots, r_n$ je univerzálna množina prvkov r_i rozptýlená v vzorkovom priestore, kde $i = 1, 2, \ldots, n$.

2) Nech $\tilde{A}$ je fuzzy množina definovaná na intervale vzorkového priestoru, ktorý obsahuje k elementov ($k < n$). Na mapovanie stupňa príslušnosti elementu r_i s fuzzy množinou $\tilde{A}$ sa označuje ako $\mu_{\tilde{A}}(r_i)$.

3) Nech $C_1, C_2, \ldots, C_m$ reprezentujú m tried, v ktorých je rozdelených n elementov.

4) Nech $S_{C_j}(r_n)$ je množina elementov triedy j na univerzálnej množine X. Je to podmnožina univerzálnej množiny X.

5) Stupeň príslušnosti elementov fuzzy množiny $\tilde{A}$ triedy j na intervale, kde $j = 1, 2, \ldots, m$ je definovaný ako:

$$D_j = \frac{\sum\limits_{r \in S_{C_j}(r_n)} \mu_{\tilde{A}}(r)}{\sum\limits_{r \in X} \mu_{\tilde{A}}(r)}. \tag{1.14}$$

6) Fuzzy Entropia $FEC_j(\tilde{A})$ elementov triedy j na intervale je definovaná ako:

$$FEC_j(\tilde{A}) = -D_j \log_2 D_j. \tag{1.15}$$

7) Fuzzy Entropia $FE(\tilde{A})$ na univerzálnej množine X pre elementy v intervale je definovaná ako:

$$FE(\tilde{A}) = \sum_{j=1}^{m} FE_{C_j}(\tilde{A}).$$

Vo výraze 1.15 je fuzzy entropia $FEC_j(\tilde{A})$ ako nepravdepodobnostná entropia. Preto môžme zadefinovať nový pojem stupeň príslušnosti pre D_j. Základná vlastnosť navrhnutej fuzzy entropie je podobná ako Shannonova entropia a spĺňa štyri Luca-Termini axiómy, ale ich spôsob merania informácie je rôzny. Pravdepodobnosť p_j Shannonovej entropie je meraná cez výskyt elementov. Oproti tomu, stupeň príslušnosti D_j v fuzzy entropii je merané príslušenstvom hodnôt vyskytujúcich sa elementov. Okrem toho, fuzzy entropia rozhodovacích oblastiach môže byť získaná cez súčet fuzzy entropie jednotlivých intervalov v každej dimenzii vlastností. [32]

Takto zadefinovaná fuzzy entropia je lepšie schopná rozlíšiť skutočné rozloženie vzorov. Použitím funkcie príslušnosti pre meranie stupňa príslušnosti, hodnota entropie obsahuje nie len počet, ale aj berie do úvahy rozloženie vzorov. [32]

Kapitola 2

Matematický popis implementovaných algoritmov

V tejto kapitole je analýza existujúcich algoritmov pre fuzzifikáciu numerických hodnôt

2.1 Algoritmus 1. Fuzzifikácia založená na fuzzy entropii

Táto sekcia popisuje fuzzy klasifikátor založený na meraní fuzzy entropie s možnosťou výberu vlastností (FEBFC).

2.1.1 Fuzzy klasifikátor založený na fuzzy entropii

Fuzzy entropia je použitá na vyhodnotenie informácie o distribúcii vzorov v priestore vzorov. S touto informáciou vedia rozdeliť priestor vzorov na disjunktné rozhodovacie oblasti pre rozoznávanie vzorov. Vďaka tomu, že rozhodovacie oblasti sú disjunktné, aj komplexnosť, aj výpočtová náročnosť je zredukovaná. Tým pádom aj čas trénovania a klasifikácie je extrémne krátka. Hoci rozhodovacie oblasti sú rozdelené do disjunktných pod priestorov, môžu dosiahnuť kvalitnú klasifikáciu vďaka tomu, že pod priestory boli správne stanovené navrhovaným meraním fuzzy entropie. Okrem toho sa skúma ďalšie využitie fuzzy entropie na vybraté prvky. Procedúra výberu prvkov nielenže znižuje

dimenziu problému, ale aj redukuje šum, zbytočné a nedôležité prvky.[32]

V klasifikačnom systéme je najdôležitejší postup rozdelenia priestoru vzoriek do rozhodovacích oblastí. Raz, keď je rozhodovacia oblasť určená, tak sa aplikujú na klasifikáciu neznámych vzorov. Rozdelenie do rozhodovacích oblastí je súčasťou je súčasťou procesu učenia či tréningového procesu, pokiaľ rozhodovacie oblastí sú rozdelené tréningovými vzormi.

V fuzzy klasifikátore založenom na fuzzy entropii sú rozhodovacie oblasti uzavreté od povrchov vytváraných z každého rozmeru. Povrchy sú určené rozložením vstupných dát. Pri vytváraní intervalov pre každý rozmer, alebo ekvivalentne, sa musí vygenerovať niekoľko trojuholníkových funkcií príslušnosti pre každú reálnu hodnotu atribútu (tento proces sa nazýva diskretizácia atribútov). [32] Počet intervalov na každom rozmere musí byť určený, ako aj centrum a šírka na každom intervale. Metóda využíva fuzzy entropiu na určenie vhodného počtu intervalov. Používa k-means algoritmus na určenie stredu intervalov. Potom ako sú určené centrá intervalov je jednoduché rozhodnúť o šírke každého intervalu.

Z vyššie uvedeného popisu je možné zhrnúť Algoritmus 1. na nasledujúce štyri kroky:

Krok A. Určenie počtu intervalov na každý rozmer.

Krok B. Určenie polohy intervalu, t.j. určenie centra a šírku pre každý interval.

Krok C. Priradenie funkcie príslušnosti pre každý interval.

Krok D. Označenie tried pre každú rozhodovaciu oblasť.

Podrobnejší popis jednotlivých krokov je popísaný v nasledujúcej časti.

2.1.2 Určenie počtu intervalov

Počet intervalov v každom rozmere má zásadný vplyv na určenie účinnosti a presnosti klasifikácie. Ak je počet intervalov príliš veľký, tak bude dlho trvať pokým sa skončí tréningový a klasifikačný proces, a môže nastať pretečenie. Na druhú stranu, ak je počet

intervalov malý, tak veľkosť každej rozhodovacej oblasti môže byť veľmi veľká, aby sa zmestila do rozdelenia vstupných vzorov, a výkon klasifikácie sa môže spomaliť. [32]

Kroky na zvolenie vhodného počtu intervalov pre každý rozmer sú:

Krok A.1 Nastavenie počiatočného počtu intervalov *I = 2*.

Krok A.2 Nájdenie centier intervalov.

Krok A.3 Priradenie funkcie príslušnosti pre každý interval.

Krok A.4 Vypočítanie celkovej fuzzy entropie pre všetky intervaly *I* a *I-1*. Počíta sa fuzzy entropia na všetkých intervalov, aby sa získala informácia o rozložení vzorov v tejto dimenzii.

Krok A.5 Klesla celková fuzzy entropia? V prípade, že celková fuzzy entropia na intervale I je menšia ako na intervaloch *I-1*, tak sa znova rozdelia *(I = I + 1)* a prejde sa na Krok A.2, inak sa prejde na Krok A.6.

Krok A.6 *I-1* je počet intervalov na zadanom rozmere. Vzhľadom na to, že fuzzy entropia neklesá, tak sa zastavilo ďalšie delenie na tejto dimenzii a *I-1* je počet intervalov na danom rozmere.

2.1.3 Určenie polohy intervalov

Proces určenia polohy intervalov začína s nájdením stredových bodov pre každý interval. Na nájdenie centier je použitý algoritmus [33, 34]. Predpokladajme, že je N počet M-rozmerných vektorov $V_i = (v_{i1}, v_{i2}, \ldots, v_{iM})^T, i = 1, 2, \ldots, N$, čo zodpovedá N prvkov. Pre rozdelenie prvkov do niekoľkých intervalov v rozmere j, sa najprv vyberie N hodnôt z prvkov reprezentujúcich tento rozmer $x_i^{(j)} = v_{ij}$. K-means zhlukovací algoritmus je použitý na klasterizáciu $x_i^{(j)} = v_{ij}$. [32]

Algoritmus určenia polohy intervaly pozostáva z nasledujúcich krokov:

Krok B.1 Nastavnie inicializačného počtu zhlukov *I*.

Krok B.2 Určenie počiatočných stredov klastrov.

Počiatočné centrá klastrov $c_1, c_2, \ldots, c_I$ môžu byť náhodne vybrané z $x_i^{(j)} = v_{ij}$. Centrá klustrov c_q ľubovolného klastra q sú priradené nasledovne:

$$c_q = \frac{q-1}{I-1}, q = 1, 2, \ldots, I.$$

Krok B.3 Priradenie označenia klastra pre každý element.

Po určení klastrových centier, sa priradí označenie pre každý prvok klastra podľa ktorého stred klastra je najbližšie. Toto je centrum s najmenšou eklidovskou vzdialenosťou od prvku. To znamená, že najbližšie centrum spĺňa nasledujúce meranie vzdialenosti:

$$\left| x_i^{(j)} - c_q^* \right| = \min_{1 \leq q \leq I} \left| x_i^{(j)} - c_q \right|$$

kde c_q^* je najbližší stred k prvku x_i^j, teda medzi $c_1, c_2, \ldots, c_I, c_q^*$ má najmenšiu euklidovskú vzdialenosť ku $x_i^{(j)}$.

Krok B.4 Prepočítanie klastrových centier.

Vzhľadom na to, že počiatočné centra sú vybrané náhode, tak sa musí prepočítať každé centrum nasledujúcim spôsobom:

$$c_q = \frac{\sum_{q=1}^{N_q} x_i^{(j)}}{N_q},$$

kde N_q je celkový počet vzorov v rovnakom zhluku q.

Krok B.5 Zmenil sa nejaký stred?

Ak každý stred zhluku je vhodne určený,tak potom prepočítanie centier v Kroku B.4 to nezmení. Ak áno, zastaví sa určovanie centier intervalov, inak sa prejde na Krok B.3 [32].

2.1.4 Priradenie funkcie príslušnosti pre každý interval

Priradenie funkcie príslušnosti je procedúra pre priraďovanie funkcie príslušnosti pre každý interval. Na aplikovanie fuzzy entropie sa zhodnotí informácia rozdelenia vzoru

v danom intervaly. Priradí sa zhodná funkcia príslušnosti pre každý interval, aby indikovala stupeň príslušnosti prvku. Hodnota v intervale môže byť videná ako stupeň elementu patriaci tomu intervalu. Intervalový stred má najvyššiu hodnotu príslušnosti, ak hodnota príslušnosti prvku klesá ako vzdialenosť medzi týmito elementami a súhlasný interval centra sa zvyšuje.Preto sa priradí najvyššia hodnota príslušnosti 1 centru intervalu, a najnižšia hodnota 0 susedom centra tohto intervalu. V tomto variante sa využíva trojuholníková fuzzy množina. Na obrázku č. 2.1 sa predpokladá, že c_1, c_2, c_3, c_4 sú centrá intervalov. Hodnoty všetkých elementov sú normalizované pre interval *[0, 1]* pre jednoduchosť. [32]

Pri priraďovaní funkcie príslušnosti intervalu, tak sa zhodnocujú tieto tri prípady:

Najľavejší interval V tomto prípade, ako je ukázané na obrázku č. 2.1, prvý centrum intervalu c_1 na tomto rozmere je ohraničený len jedným intervalovým centrom c_2. Najvyššia hodnota príslušnosti 1 tohto intervalu sa nachádza v c_1, kde je najnižšia hodnota príslušnosti 0 je v c_2. Keď $x = 0$, hodnota príslušnosti je určená ako 0.5, ako je ukázané na obrázku č. 2.1. Funkcia príslušnosti μ_{i1} najľavejšieho intervalu na rozmere i je definovaná nasledovne:

$$\mu_{i,1}(x) = \begin{cases} \frac{c_1+x}{2c_1}, & \text{pre } x \leq c_1, \\ \max\left\{1 - \frac{|x-c_1|}{|c_2-c_1|}\right\}, & \text{pre } x > c_1, \end{cases}$$

kde c_1 je centrum najľavejšieho intervalu, a c_2 je centrum prvého centra intervalu vpravo od c_1.

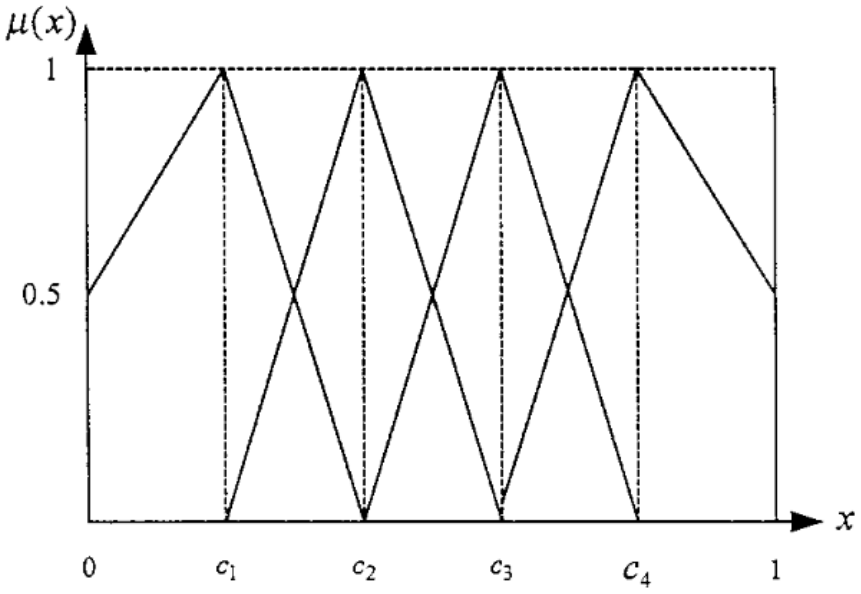
Najpravejší interval V tomto prípade, ako je ukázané na obrázku č. 2.1, funkcia príslušnosti μ_{i4} najpravejšieho intervalu na rozmere i je definovaný nasledovne:

$$\mu_{i,4}(x) = \begin{cases} \max\left\{1 - \frac{|c_4-x|}{c_4-c_3}, 0\right\}, & \text{pre } x \leq c_4, \\ \frac{2-x-c_4}{2(1-c_4)}, & \text{pre } x > c_4, \end{cases},$$

kde c_4 je centrom najpravejšieho intervalu, a c_3 je centrom prvého intervalu vľavo od c_4.

Interný interval V tomto prípade, ako je ukázané na obrázku č. 2.1, center c_3 interného intervalu je ohraničený jeho ľavým intervalovým centrom c_2 a pravým intervalovým centrom c_4. Najvyššia hodnota príslušnosti sa nachádza v c_3, a najnižšie hodnoty sú v centrách c_2 a c_4. Funkcia príslušnosti μ_{i3} je tretím intervalom v rozmere i v tomto prípade definovanom ako

$$\mu_{i,3}(x) = \begin{cases} \max\left\{1 - \frac{|c_3 - x|}{|c_3 - c_2|}, 0\right\}, & \text{pre } x \leq c_3, \\ \max\left\{1 - \frac{|c_3 - x|}{|c_4 - c_3|}, 0\right\}, & \text{pre } x > c_3. \end{cases}$$



Obr. 2.1: Príklad priradenia funkcie príslušnosti pre intervalové centrá c_1, c_2, c_3, c_4 a trojuholníky korešpondujúce s funkciou príslušnosti[32].

2.1.5 Označenie tried pre každú rozhodovaciu oblasť

V tomto kroku sa vypočíta fuzzy entropia pre každú vlastnosť cez sumarizáciu fuzzy entropie vo všetkých intervalov v tomto rozmere vlastnosti.

Pre označenie tried v každej rozhodovacej oblastí sa musí použiť metóda na určenie fuzzy entropie. Fuzzy entropia rozhodovacích oblastí pre vzory každej triedy sú počítané tak, aby sa určila trieda pre každú rozhodovaciu oblasť. Fuzzy entropia rozhodovacích

oblastí môže byť získaná cez sumarizáciu fuzzy entropie individuálnych intervalov pre každú vlastnosť rozmeru. Triede sa priradia rozhodovacia oblasť s najnižšou fuzzy entropiou v tejto oblasti. Raz keď je rozhodovacia oblasť priradená a trieda označená, tak trénovanie je kompletné[32].

2.1.6 Výber vlastností

Nový prístup pre výber funkcie založenej na fuzzy entropii je popísaný v nasledujúcom odseku.

Fuzzy entropia odráža viac informácií v aktuálnom rozložení vzorov v priestore vzoriek. Vzhľadom k tomu, že fuzzy entropia je schopná rozlíšiť rozdelenie vzorov lepšie, používa sa na hodnotenie oddeliteľnosti jednotlivých funkcií. Intuitívne, keď je nižšia fuzzy entropia vlastností, tak tým vyššie je znevýhodnenie vlastností.[32].

Akonáhle je určená fuzzy entropia pre každú vlasnosť, tak sa môžu vyberať vlastností podľa výberu dopredu (*forward selection*) alebo dozadu (*backward elimination*). Metóda výberu dopredu je určenie relevantných vlastností na začiatku s prázdnou množinou a iteratívne pridávať vlastností až pokým nie sú splnené kritérium na zastavenie. Na rozdiel od toho eliminačná metóda vzad začína so všetkými vlastnosťami v množine, a odstraňuje vlastností pokým sú splnené kritérium na zastavenie. V Algoritme 1. sa používa eliminačná metóda vzad na selektovanie relevantných vlastností. Kritérium zastavenia v tejto metóde je založená na klasifikácii rýchlosti triediča Vzhľadom k tomu, že vlastností s vyššou fuzzy entropiou sú málo relevantné pre klasifikačný cieľ, tak sa odstránia vlastnosti ktoré majú najvyššiu fuzzy entropiu, ak to nezníži mieru klasifikácie. Potom sa opakuje vyššie spomenutý krok až pokým všetky irelevantné vlastností sú odstránené. Napokon ľavé vlastností sú určené ako vlastností pre klasifikáciu. S touto selekciou vlastností sa môže znížiť problém s rozmerom, aby sa urýchlil proces klasifikácie. V niektorých prípadoch sa môžu dosadnúť lepšie výsledky klasifikácie tým, že odhalíme nadbytočné, šumivé alebo nedôležité vlastnosti[32].

2.1.7 Zhrnutie Algoritmu 1.

Cieľom tradičnej klasifikácie vzorov je rozdelenie priestoru vzoriek do rozhodovacích oblastí a to jednu oblasť pre každú triedu. V mnohých klasifikačných systémoch sú práve rozhodovacie oblasti rozdelené do prekrývajúcich sa oblastí. Hoci klasifikátor s prekrývajúcimi rozhodovacími oblastiami môže dosiahnúť lepšiu klasifikačnú výkonnosť, ale to trvá dlhšie pre uzatvorenie rozhodovacích oblastí[32].

Táto metóda je prezentovaná ako efektívny klasifikátor s výberom vlastností založený na fuzzy entropii pre klasifikáciu vzoru. Vzorový priestor je rozdelený do neprekrývajúcich fuzzy rozhodovacích oblastí. Vzhľadom na to, že rozhodovacie oblastí sú fuzzy pod-oblasti, tak sa môžu získať hladké hranice, pre dosiahnutie lepšieho výkonu klasifikácie. Aj keď rozhodovacie oblastí sa neprekrývajú, tak môžu znížiť výpočtovú zložitosť a záťaž klasifikátora[32].

Tiež sa používa fuzzy entropia na vybranie relevantných vlastností. Aplikovanie výberu vlastností nielen zníži problém s rozmerom, ale ako aj zlepší výkonnosť klasifikácie odstránením zbytočných, šumivých, nedôležitých vlastností. Taktiež algoritmus K-means zhlukovania bol použitý na určenie funkcie príslušnosti pre každú vlasnosť[32].

2.1.8 Algoritmus 2. Modifikácia - Hierarchická fuzzy entropia

Táto sekcia popisuje fuzzy klasifikátor založený na hierarchickej fuzzy entropii (FC-HFE), ktorý vychádza z Algoritmu 1.

Algoritmus 1. má nasledovné problémy:

- Označovanie tried pre každú rozhodovaciu oblasť.
- Rastúci počet intervalov na každej dimenzii.
- Neprekrývajúce sa rozhodovacie oblasti.

Označovanie tried pre každú rozhodovaciu oblasť je vykonávané podľa fuzzy entropie súčtom individuálnych intervalov pre každú dimenziu. Rozhodovacia oblasť je priradená triede s najnižšou fuzzy entropiou v danej oblasti. Podľa axiómu fuzzy entropie, fuzzy entropia bude nula, napriek tomu či stupeň príslušnosti bude pre každú triedu nula alebo

jedna. Priraďovanie tried bude mať chyby, keď počet tried v ktorých fuzzy entropia sa rovná nule je viac ako jeden v tejto oblasti[35].

Rastúci počet intervalov na každej dimenzii je vykonaný podľa pravidla,pre ktoré celková entropia *I* intervalov je menej ako to pre *I-1* intervalov. Niekedy práve toto pravidlo zabraňuje v raste počtu intervalov, ktoré vedie k nesprávnej klasifikácii. Napríklad, dáta sú prezentované ako párne pretínajúce rozdelenie, napríklad dáta tvaru špirály. Fuzzy entropia špirálových dát tohto typu je na začiatku vysoká, dáta by mali byť neskôr rozdelené na základe klesajúcej fuzzy entropie. Preto podmienka na zastavenie algoritmu pre rastúci počet intervalov bude výsledok v obmedzení rastúceho počtu intervalov na každej dimenzii[35].

Vzorový priestor bol rozdelený do neprekrývajúcich sa rozhodovacích oblastí použitím mriežkového rozdelenia. Označovanie tried nemôže byť priradené do rozhodovacej oblasti, ktorá nemá žiadne trénovacie dáta. Ak testovacia vzorka padá do danej rozhodovacej oblasti, ktoré systém nevie ako klasifikovať[35].

Modifikácia upravuje pôvodný algoritmus v určovaní počtu intervalov napríklad udržiava pôvodnú podmienku pre zastavenie rastu počtu intervalov, ak celková entropia *I* intervalov je viac ako jeden z *I-1* intervalov. Ďalšia podmienka pre zastavenie rastu počtu intervalov je, ak celková fuzzy entropia *I* intervalov je viac ako jedna z *I-1* intervalov a celková fuzzy entropia *I-1* intervalov je menej ako prahová hodnota φ. To je, keď celková fuzzy entropia *I* intervalov je menšia ako jedna z *I-1* intervalov. Alebo celková entropia *I* intervalov je viac ako prahová hodnota φ, tak $I = I + 1$.

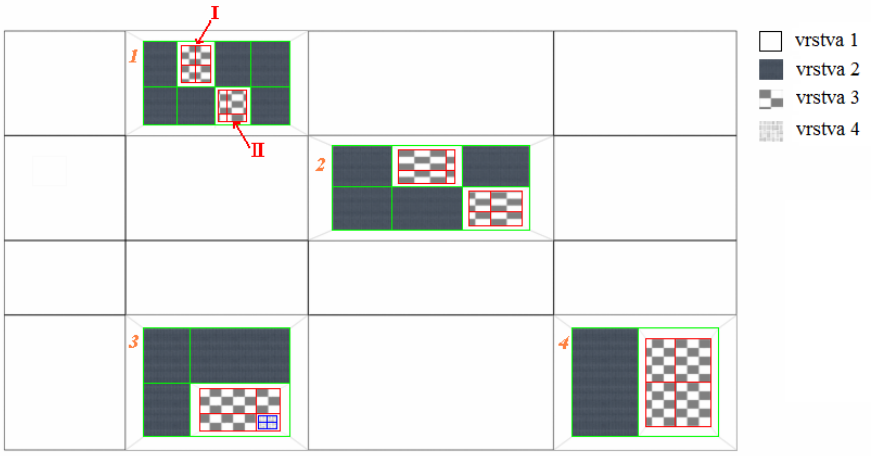
Hodnota prahovej hodnoty φ bola vypočítaná použitím nasledovnej rovnice:

$$\varphi = -N_{trieda} * \frac{1}{N_{trieda}} * \log_2 (\frac{1}{N_{trieda}}) * (I-1) * \theta = -\log_2 (\frac{1}{N_{trieda}}) * (I-1) * \theta$$

kde N_{trieda} je počet tried s hodnotami, a θ je percento maximálnej celkovej fuzzy entropie z *I-1* intervalov. θ môže byť ladený použitím odlišného klasifikačného problému. Preto, môžu produkovať dostatok intervalov pre rovnomerné preniky dáta distribúcii použitím tejto metódy. Fuzzy entropia pre každú dimenziu musí byť menej ako špecifická hodnota.

Ďalší fenomén je hodnota fuzzy entropie pre niektoré rozhodovacie oblasti je vždy veľmi vysoká. Rozdelenie dát rozhodujúcej oblasti je veľmi nejednoznačné alebo dáta je veľmi veľa nesprávne klasifikované dát. Vzorkový priestor je rozdelený do toľko rozhodovacích oblastí fuzzy entropiou, zabraňujúc narastaniu klasifikačnej hodnote.

Modifikácia sa zaoberá vyššou fuzzy entropiou rozhodovacích oblastí nazývaných hierarchická fuzzy entropia. Štruktúra je zobrazená na obrázku 2.2. 2D vzorkový priestor na obrázku 2.2 je rozdelený do nerovnomerných 53 rozhodovacích oblastí hierarchickou fuzzy entropiou. Prvá až štvrtá rozhodovacia oblasť fuzzy entropie vrstvy *I* je veľmi vysoká. Rozdelili sa tieto rozhodovacie oblasti. Napríklad rozhodovacia oblasť *I* bola rozdelená do ôsmych rozhodovacích oblastí po druhý raz. Rozhodovacie pod oblasti *I* a *II* boli rozdelené do štyroch rozhodovacích oblastí retrospektívne po tretí raz. Rozhodovacie oblasti *I* boli rozdelene do štrnástich rozhodovacích oblastí. Fuzzy entropia väčšiny rozhodovacích oblastí sa stala nižšou po tom, ako bol vzorkový priestor rozdelený použitím hierarchickej fuzzy entropie. [35].



Obr. 2.2: Pohľad zhora na rozdelené rozhodovacie oblasti použitím hierarchickej entropie [35].

Okrem toho, je modifikovaná metóda, ktorou sa počíta fuzzy entropia pre rozhodovacie oblasti. Ak tam nie sú žiadne trénovacie dáta určitej triedy na intervale, tak množina fuzzy entropie tejto triedy sa rovná jednej. To je preto, aby sa zabránilo tomu, že fuzzy entropia určitej triedy sa bude rovnať nule viac ako jedenkrát pre počítanie najnižšej

fuzzy entropie oblasti.

Algoritmus 2. má tieto nasledujúce kroky [35].:

Krok 1 – Krok 4. je rovnaký ako v Algoritme 1. v selektovaní počtu intervalov pre každú dimenziu na aktuálnom vzorovom priestore.

Krok 5. Ak celková fuzzy entropia I intervalu je menej ako I-1 intervalu alebo celková fuzzy entropia I intervalu je viac ako prahová hodnota φ, tak sa znova rozdeľuje $(I = I - 1)$ a ide na krok 2. Inak sa zastaví zvyšovanie intervalov na tejto dimenzii a rozhodne sa počet intervalov pre ďalšiu dimenziu.

Krok 6. Akonáhle pre každú dimenziu sú intervaly určené, rozhodovacie oblasti pre súčasný vzorkový priestor sú rozdelené. Stredná hodnota fuzzy entropie je vypočítaná použitím nasledujúcej rovnice pre všetky rozhodovacie oblasti:

$$MFE_i = \frac{\sum_{i=1}^{N_i} FE_{ij}}{N_i}, j = 1 \ldots N_i,$$

kde N_i je počet rozhodovacích oblastí pre i-tú vrstvu, FE_{ij} je j-tá rozhodovacia oblasť pre i-tú vrstvu a MFE_i je stredná hodnota fuzzy entropie pre i-tú vrstvu.

Krok 7. Fuzzy entropia pre každú rozhodovaciu oblasť v súčasnom vzorovom priestore je porovnávaná s *MFE*. Ak fuzzy entropia pre rozhodovaciu oblasť je väčšia ako stredná hodnota fuzzy entropie, oblasť sa stane nezávislým pod priestorom a Krok 1. až Krok 7. sú vykonávané znova na určenie rozhodovacích oblastí pre každý pod priestor. Inak rozhodovacia oblasť bude pridelená označeniu triedy.

Horeuvedené kroky sú ako rekurzívna funkcia, ktorá je vykonávaná opakovane pre každý pod priestor až pokiaľ všetky pod priestory nemôžu byť viac rozdelené. Keď kroky sú skončené, testovací vzor je klasifikovaní použitím rozhodovacích oblasti, ktorá majú označenie triedy. Keď testovacia vzorka nemá označenie triedy oblasti triedy, tak testovacia vzorka je porovnávaná použitím krátkej euklidovskej vzdialenosti medzi susedmi trénovaných dát a testovacej vzorky, klasifikované do najbližšej trénovanej triedy.[35].

2.1.9 Algoritmus 3. Modifikácia - Vážená entropia

Táto sekcia popisuje fuzzy klasifikátor založený na váženej entropie Algoritmus 3. sa od Algoritmu 1. odlišuje v spôsobe výpočtu funkcie príslušnosti lingvistickej premennej, tak aby hodnota sumy hodnôt funkcie príslušnosti bola rovná jednej. Ďalej pri výbere kritéria efektívnosti rozkladu na intervaly v tvare váženej fuzzy entropie, ktorá berie do úvahy aj počet elementov patriacich do získaných intervalov. Základná myšlienka spočíva v postupnom rozklade množiny X na $2,\ldots,n$ intervalov a kontrole efektívnosti tohto rozkladu. [1]

Vstupné dáta

Počet N hodnôt vstupného atribútu, zadaného množinou reálnych čísel $X = x_1,\ldots,x_i,\ldots,x_N$. Počet K možných hodnôt výstupného atribútu $B = b_1,\ldots,b_k,\ldots,b_K$, ku ktorým patria prvky množiny X. Množina dvojíc (x_i, b_k) definuje vzťah medzi každou hodnotou množiny X a hodnotou výstupného atribútu. [1]

Výstupné dáta

Počet Q intervalov, na ktoré je potrebné rozložiť vstupnú množinu reálnych čísel X (počet termov lingvistickej premennej). Matica príslušnosti U s rozmermi $N \times Q$. [1]

Kroky algoritmu

Algoritmus 4. sa skladá z týchto ôsmych krokov:

Krok 1. Určenie počiatočného počtu intervalov Q = 2.

Krok 2. Vybratie náhodného počiatočného centra každého intervalu $I = \left\{C_1,\ldots,C_Q\right\}$.

Krok 3. Určenie intervalu, kde patrí každý element x_i. Kritérium je najmenšia hodnota euklidovskej vzdialenosti.

Krok 4. Nové centrá I sú určené pre každý interval ako aritmetický priemer hodnôt

prvkov x_i, patriacich do týchto intervalov

$$C_q = \frac{\sum_{i=1}^{N_q} x_i}{N_q},$$

kde N_q je počet prvkov x_i patriacich do intervalu C_q.

Krok 5. Ak sa jedno z vypočítaných nových centrier zmenilo, tak sa pokračuje Krokom 3, v opačnom prípade Krokom 6.

Krok 6. Definícia funkcii príslušnosti lingvistickej premennej pre každý z Q termov. Získaný výsledok predstavuje hľadanú maticu príslušnosti **U**.

Krok 7. Vypočítanie váženej fuzzy entropie $wFE(A)_Q$ pri rozklade na Q intervalov.

Krok 8. Vypočítaná fuzzy entropia $wFE(A)_Q$ sa porovná s predchádzajúcou hodnotou $wFE(A)_{Q-1}$. Pričom prvá hodnota je veľké číslo $wFE(A)_1 = +\infty$. V prípade, že táto entropia neklesá, tak sa zväčší počet intervalov $Q = Q+1$ a zopakujú sa všetky výpočty vrátane Kroku 2. V opačnom prípade hodnota $(Q-1)$ je optimálny počet intervalov. [1]

2.2 Algoritmus 3. Modifikácia s Fuzzy k-means algoritmom

Táto modifikácia namiesto klasického k-meansu algoritmu použije FCM algoritmus na určenie polôh centier v intervaloch.

Výhody FCM algoritmu sú, že dáva najlepšie výsledky pre prekrývajúce sa súbory dát a pomerne lepšie ako k-means algoritmus. Na rozdiel od k-means, kde dátový bod musí výlučne patriť do jedného centra klastra, v tomto dátovom bode je priradené členstvo v každom klastrovom centre, v dôsledku čoho môže byť dátový bod patriť do viac ako jedného centra klastra.

Nevýhody sú, že apriori sa definuje počt zhlukov. S nižšou hodnotou beta hodnoty dostaneme lepší výsledok, ale na úkor väčšieho početu iterácií. Euklidovské meranie vzdialenosti môže nerovnomerne zaťažiť základné faktory.

Fuzzy k-means (FCM) je metóda, ktorá umožňuje zhlukovanie pre každý dátovy bod, ktorý môže patriť do viacerým klastrov s rôznymi stupňami príslušnosti. [36]

FCM metóda je založená na minimalizácii účelovej funkcie

$$J_m = \sum_{i=1}^{D}\sum_{j=1}^{N} \mu_{ij}^m \left|\left|x_i - c_j\right|\right|^2,$$

kde D je počet dátových bodov, N je počet klastrov, m je fuzzy časť exponentu matice pre riadenie stupňa fuzzy prekrytia, m > 1. Fuzzy prekrytie určuje ako rozmanité hranice sú medzi klastrami a aký počet dátových bodov majú významnú príslušnosť vo viac ako v jednom klastri. Ďalej x_i je i-tý dátový bod, c_j je centrum j-tého klastra, μ_{ij} je stupeň príslušnosti x_i v j-tom klastri. [37]

Kroky Fuzzy k-means algoritmu :

Krok 1. Náhodne sa inicializujú klustre hodnoty príslušnosti, μ_{ij}.

Krok 2. Vypočítanie centier klastra $c_j = \frac{\sum_{i=1}^{D} \mu_{ij}^m x_i}{\sum_{i=1}^{D} \mu_{ij}^m}$

Krok 3. Aktualizácia μ_{ij} podľa nasledujúceho vzťahu $\mu_{ij} = \frac{1}{\sum_{k=1}^{N} \left(\frac{\left|\left|x_i - c_j\right|\right|}{\left|\left|x_i - c_k\right|\right|} \right)^{\frac{2}{m-1}}}$.

Krok 4. Vypočítanie účelovej funkcie J_m.

Krok 5. Opakovanie krokov 2-4 až pokým J_m sa zlepší o menej než stanovená minimálna prahová hodnota alebo po uplynutí určitého maximálneho počtu iterácií.

2.3 Algoritmus 4. Vážená entropia s FCM algoritmom

Kroky algoritmu sú tie isté ako v algoritme 1, ale pri podmienke na zastavenie sa bude brať vážená entropia. Na lokáciu centier intervalov sa použije FCM algoritmus.

Kapitola 3

Implementácia nástroja pre fuzzifikáciu hodnôt

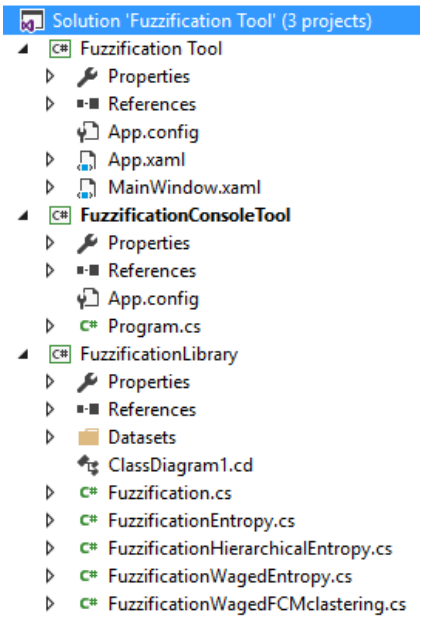
Nástroj je implementovaný v jazyku C++ a C#. V jazyku C++ sú naprogramované spomenuté algoritmy v práci. V C# som urobila konzolovú aplikáciu, a grafické rozhranie pre implementované algoritmy. Výstupmi programu sú textové súbory s výsledkami a informáciami o daných dátach.

3.1 Analýza a návrh

Základná funkcionalita nástroja je spracovanie reálnych dát na fuzzy hodnoty. V programe je už vopred zadefinovaná štruktúra dát a počet výstupných atribútov. Následne pomocou metódy fuzzifikácie sa transformujú dané hodnoty na fuzzy hodnoty. Užívateľské rozhrani má vopred definovanú množinu dát. Vstupy a výstupy aplikácie sú tom istom priečinku ako program.

Grafické uživateľské rozhranie a konzolovú aplikáciu som implementovala v jazyku C#, hlavné kvôli grafickým prvkom WPF technológie a prípadnému využitiu C++ .DLL knižníc.

Nástroj bol implementovaný vo vývojom prostredí Visual Studio. Štruktúru C# projektu je zobrazená na obrázku č. 4.2.



Obr. 3.1: Štruktúra projektu pre nástroj Fuzzy Tool

3.2 Implementácia nástroja fuzzy tool

Na spracovanie dátových množín som navrhla triedu *DataSets*, ktorá bude mať potomkov pre každý jeden druh datasetu (DatasetHeart, DatasetIris...). Na fuzzifikovanie dát som navrhla hlavnú triedu *Fuzzification*, ktorá má potomkov *FuzzificationEntropy*, *FuzzificationHierarchicalEntropy*, *FuzzificationWagedEntropy* a *FuzzificationWagedFCMclastering*. V triede *Fuzzification* budem využívať konkrétnu inštanciu DataSets. Na UML diagrame č. 3.2 sú zobrazené triedy s metódami a atribútmi a ich závislosti.

Na obrázku číslo 3.2 je zobrazený diagram tried implementovaného fuzzification nástroja. Jednotlivé triedy a funkcionalitu opíšem v nasledujúcej časti.

3.2.1 Množiny dát - Datasets

Na zadefinovanie vlastností súboru dát som použila nasledovné properties (v C++ get metódy):

- *Attributes* - počet vstupných a výstupných atribútov v dátovom súbore.
- *InputAttributes* - počet vstupných atribútov v dátovom súbore.

- *OutputAttributes* - počet výstupných atribútov, hodnota je 1.
- *OutputIntervals* - počet intervalov výstupného parametra.
- *LingvisticAttribute* - hodnoty lingvistického atribútu.
- *DatasetSize* - počet všetkých prvkov v dátovej množine.
- *Filename* - názov vstupného súboru.
- *InitialError* - počiatočná chyba, ktorá je vypočítaná po načítaní súboru.
- *Dataset* - normalizované hodnoty datasetu v intervale $< 0, 1 >$.

Medzi hlavnú funkcionalitu triedy patrí metóda *InitializeDataset*, ktorá je abstraktná a je prekrytá v jej potomkoch. Metóda vykonáva načítanie dát zo súboru, normalizovanie súboru a vypočítanie počiatočnej chyby.

Potomkovia využívajú metódy triedy *DataSets* a to sú tieto:

- *ClearDatasets* - vymazanie údajov.
- *ComputeInitialError* - vypočítanie počiatočnej
- *NormalizeDataset* - normalizácia dát do intervalu od nula po jedna.
- *RandomizeArray* - náhodne poprehadzuje poradie dát v dátovej množine.
- *ShrinkDataset* - zmenší dáta o zadané percento.
- *ToString* - všetky informácie v reťazci.
- *WriteInfoToFile* - vypísanie podrobných informácií o datasete do súbora so zadaným názvom.
- *WriteToFile* - vypísanie normovanej dátovej množiny do súboru.

Diagram tried potomkov Datasetu je znázornený na obrázku č.3.3

Spracovanie dát

Inicializácia dát sa vykonáva vytvorením inštancie potomka textitDataSets. Táto inštancia je parametrom konštruktora abstraktnej triedy *Fuzzification*, kde sa inicializuje do atribútu *DataToTransform*. *Centers* sú centrá všetkých atribútov dát v intervaloch. Počty intervalov pre jednotlivé dimenzie sú v poli *Intervals*. Výsledná množina fuzzifikovaných dát sa nachádza v poli *Results*. Celková entropia pre jednotlivé intervaly je v poli *TotalEntropy*.

Fuzzification trieda má zadefinovanú metódu *Initialize*, v ktorej inicializuje všetky atribúty na defaulné hodnoty. *RunFuzzification* je najdôležitejšia metóda triedy, pretože v nej sa spúšta celý proces transformácie numerických hodnôt na lingvistické. V tejto metóde sa inicializuje dátová množina, a pre všetky lingvistické atribúty sa nainicializuje ich hodnota do vopred určených intervalov. Následne sa pre všetky atribúty, ktoré nie sú lingvistické spustí proces transformácie reálnych dát na lingvistické. Pre každú dimenziu (t.j. atribút) sa vykoná metóda *RunFuzzificationInDimension*.

Algorithm 1 Algoritmus *RunFuzzificationInDimension* vykonáva nasledovné kroky:

```
int interval = SetInitialNumberOfIntervals(dimension);
TotalEntropy[dimension][i] = 99999999999;
bool condition = false;
while (!condition) do
    ResizeResultToNewInterval(dimension, interval);
    Centers[dimension] = DeterminationLocation(dimension, intervals);
    MembershipFunctionAssignment(dimension, interval);
    TotalEntropy[dimension][interval] = ComputeEntropy(dimension);
    condition = ConditionForStoping(dimension, TotalEntropy[dimension][interval],
    TotalEntropy[dimension][interval - 1]);
    interval++;
end while
LastStepInFuzzification(dimension, interval);
```

Na určenie polôh jednotlivých stredov intervalov je použitá abstraktná metóda. V

prípade potomka *FuzzificationEntropy* je počítaná klasickým K-Means algoritmom, a u potomka *FuzzificationWagedFCMclastering* je pomocou algoritmu FCM.

MembershipFunctionAssignment je virtuálna metóda určená na výpočet hodnôt príslušností dát v určitom intervale.

Metóda na výpočet celkovej funkcie je abstraktná a v prípade potomka *FuzzificationEntropy* sa vypočíta celková a pri potomkovi *FuzzificationWagedEntropy* vážená entropia.

V metóde *LastStepInFuzzification* sa zníži počet intervalov a prepočítajú sa centrá a hodnoty funkcie príslušnosti. V prípade potomka *FuzzificationHierarchicalEntropy* sa prekryje daná metóda,aby sa neznížil počet intervalov o jedna v poslednom kroku fuzzifikácie.

Na zápis výsledkov a informácií o priebehu fuzzifikácie sú použité metódy WriteToFile, a WriteResultsToFile.

Vzťahy a prepojenia jednotlivých potomkov triedy sú znázornené na obrázku č. 3.4.

3.3 Ovládanie nástroja používateľom

3.3.1 Vstupné údaje programu

Forma vstupných dát je vopred definovaná (dáta sú oddelené, buď čiarkov alebo iným oddeľovačom). Dáta, ktoré sa majú fuzzifikovať sú v priečinku spolu s programom.

3.3.2 Užívateľská príručka

Ovládanie nástroja sa skladá z nasledujúcich krokov:

1. Užívateľ spustí aplikáciu. Zobrazí sa mu menu, kde vyberie dátovú množinu a algoritmus fuzzifikácie. Menu konzolovej aplikácie je zobrazené na obrázku č. 3.5 a menu grafického rozhrania je na obrázku č. 3.6.

2. Spustenie fuzzifikácie sa vykoná zadaním čísla jeden, a v prípade grafického rozhrania je to stlačenie tlačidla na fuzzifikovanie dát.

3. Čakanie na výsledky. V prípade konzolovej aplikácie sa zobrazí aktuálna hodnota celkovej fuzzy entropie a počet tried v danom intervale pre konkrétny atribút.

4. Úspešne fuzzifikovanie dát. Užívateľ je upozornení o úspešnom ale aj neúspešnom vykonaní fuzzifikácie. V prípade konzolovej aplikácie je to správa vypísaná na konzolu, a pre grafické rozhranie je to vyskakovacie okno, ktoré je znázornené na obrázku č. 3.7.

5. Ďalšie operácie. V prípade konzolovej aplikácie užívateľ zadá ďalšie dodatočné operácie. Výpis výsledkov do súboru (voľba 4), výpis informácií o dátovej množine (voľba 2) a podrobnejšie výsledky fuzzifikácie (voľba 3) do súboru.

6. Súbory. Užívateľ nájde výsledné súbory v priečinku, ako je znázornené na obrázku č. 3.8.

3.3.3 Opis výstupných súborov programu

Informácie o vstupnom súbore

V súbore *dataset-information* sú zobrazené základné informácie o súbore dát, ktoré boli fuzzifikované. Obsahuje počet dát, vstupných atribútov, výstupných atribútov, výstupných intervalov, počet intervalov lingvistických dát, názov súboru, počiatočnú chybu a normované dáta.

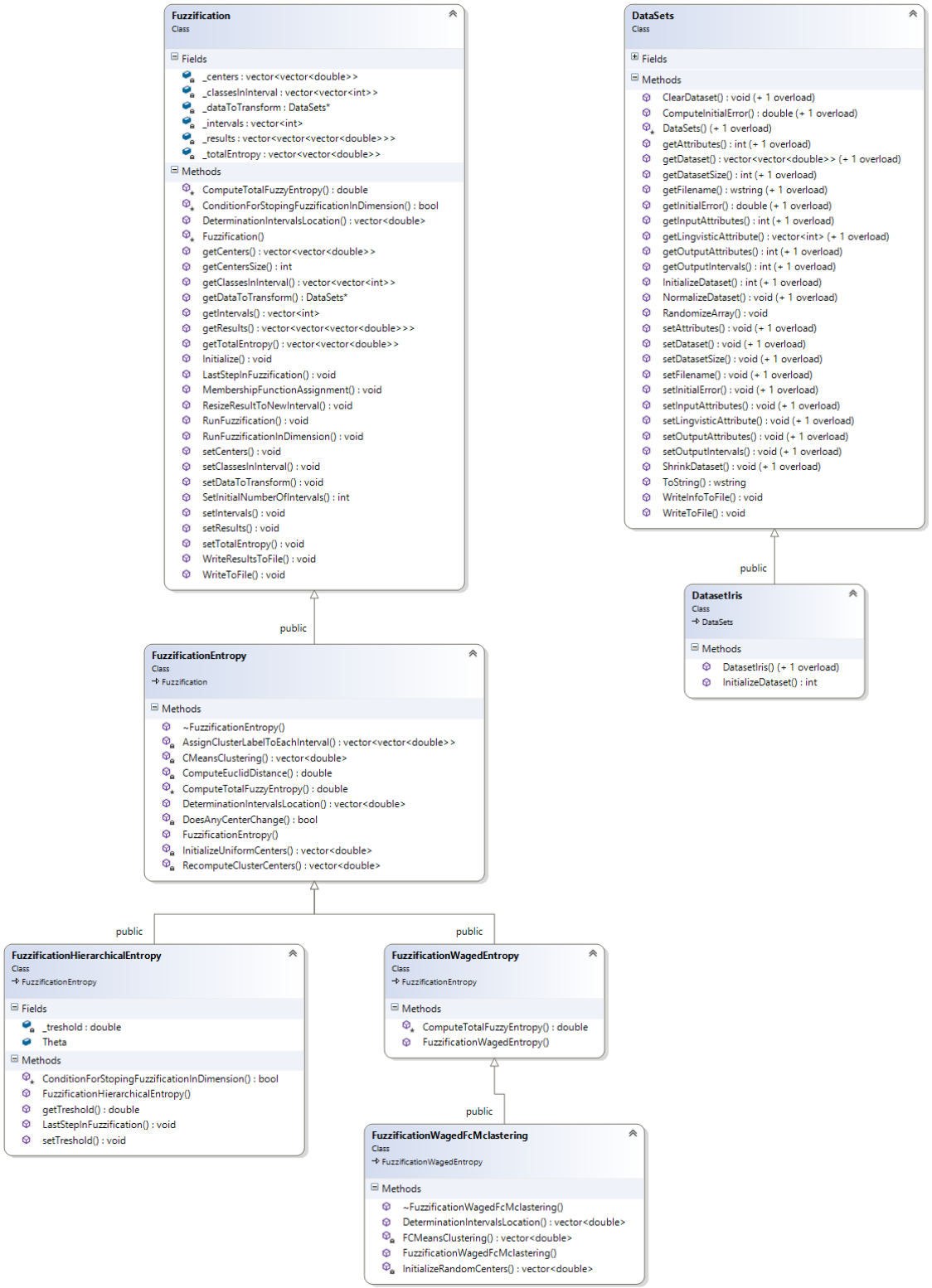
Výsledky fuzzifikácie

Výsledky fuzzifikácie sú zapísané do súboru *fuzzification-results*. Obsahuje lingvistické dáta v rozsahu od nula po jedna, a súčet intervalu pre konkrétny atribút dáva hodnotu jedna. Príklad obsahu súboru je znázornený na obrázku č.3.9.

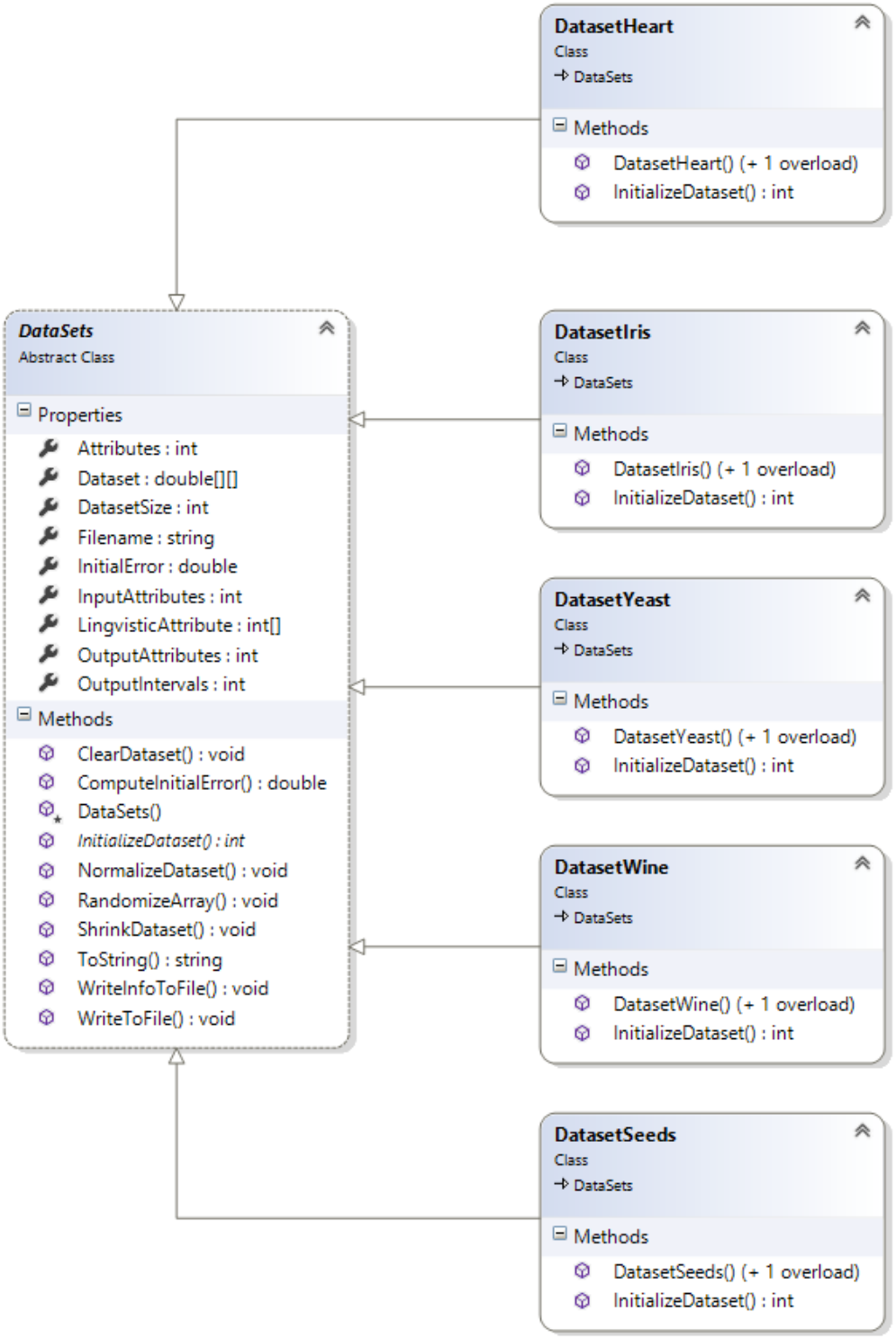
Podrobnejšie výsledky fuzzifikácie

Podrobnejšie výsledky sú zapísané do súboru *fuzzification-information*. Obsahuje údaje celkovej entropie pre jednotlivé dimenzie na každom intervale. Ďalej obsahuje počet

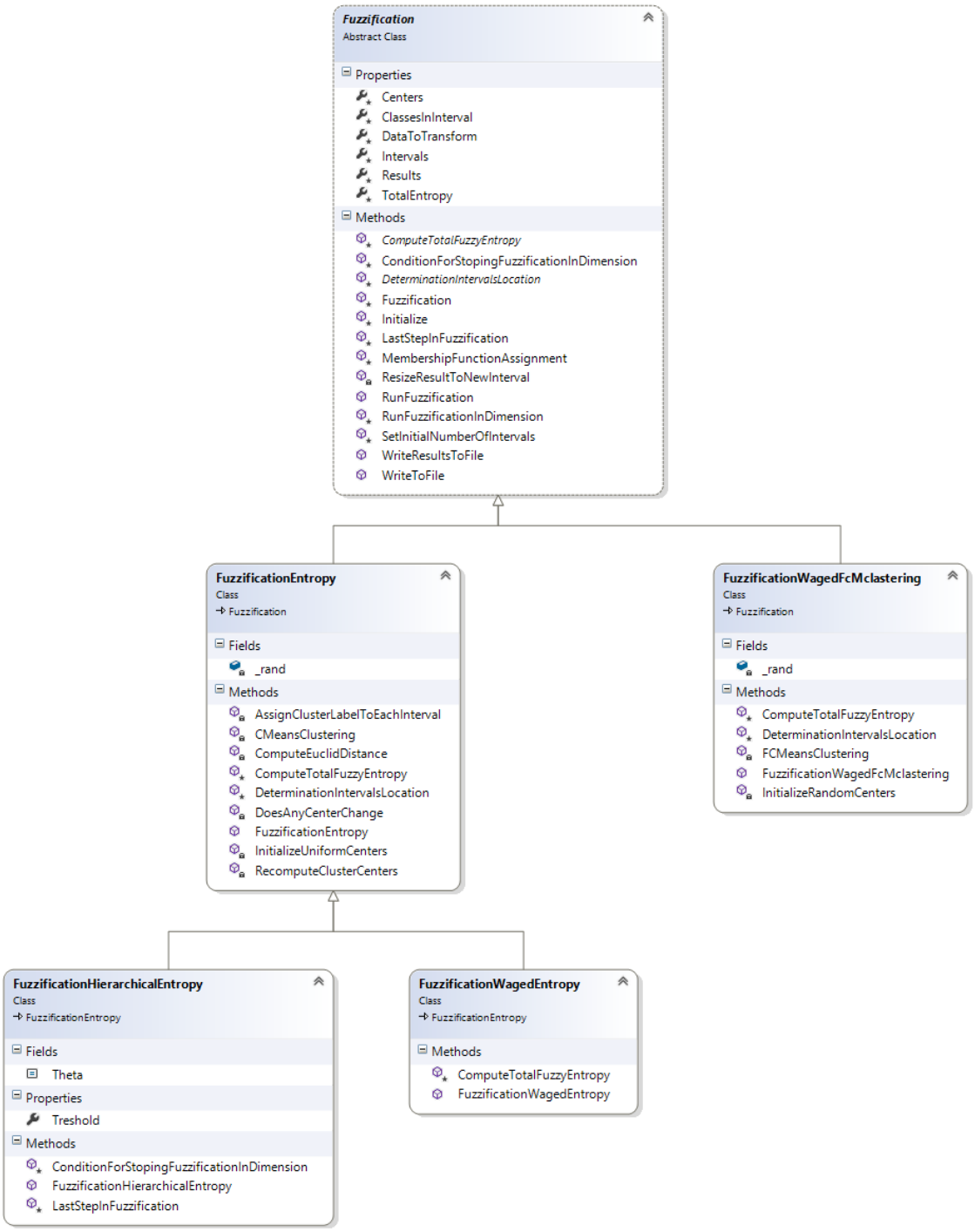
intervalov pre každú dimensiu a ako aj lokáciu jednotlivých centier. V súbore sa nachádzajú aj počty výstupných tried čo sa nachádzajú na danom atribúte. Na konci súboru sú výsledky fuzzifikácie spolu s kontrolným súčtom jedna. Príklad obsahu súboru je znázornený na obrázku č. 3.10.



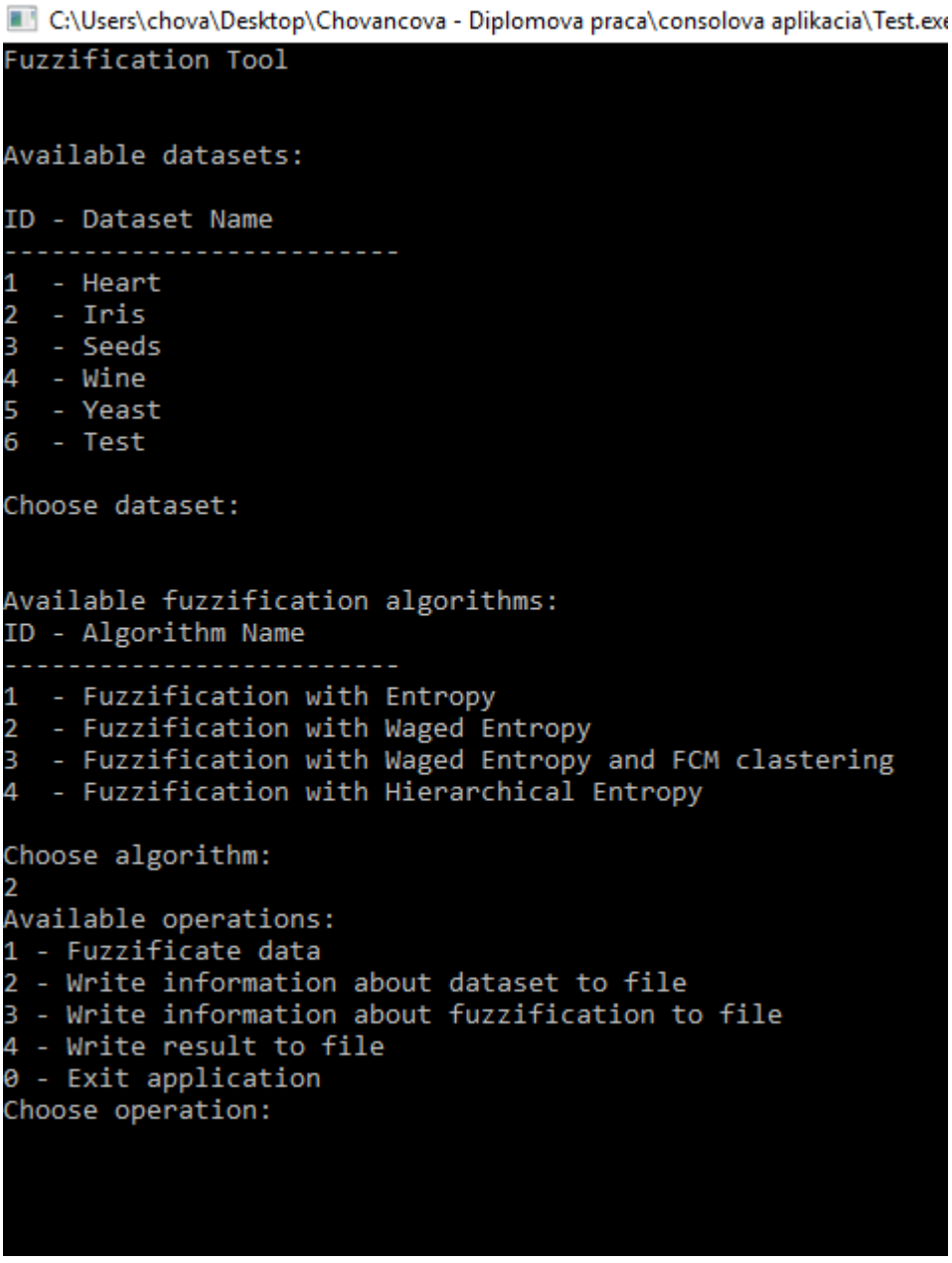
Obr. 3.2: Diagram tried v C++



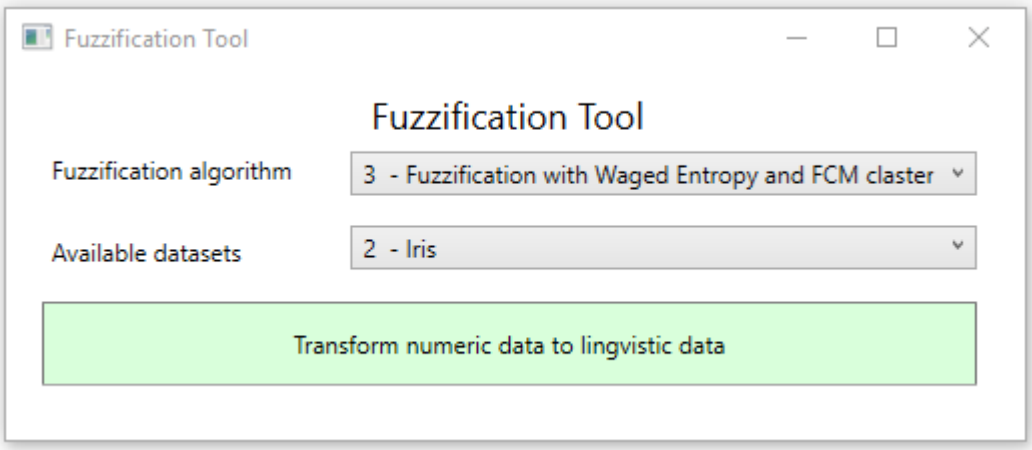
Obr. 3.3: Diagram tried potomkov Datasetu



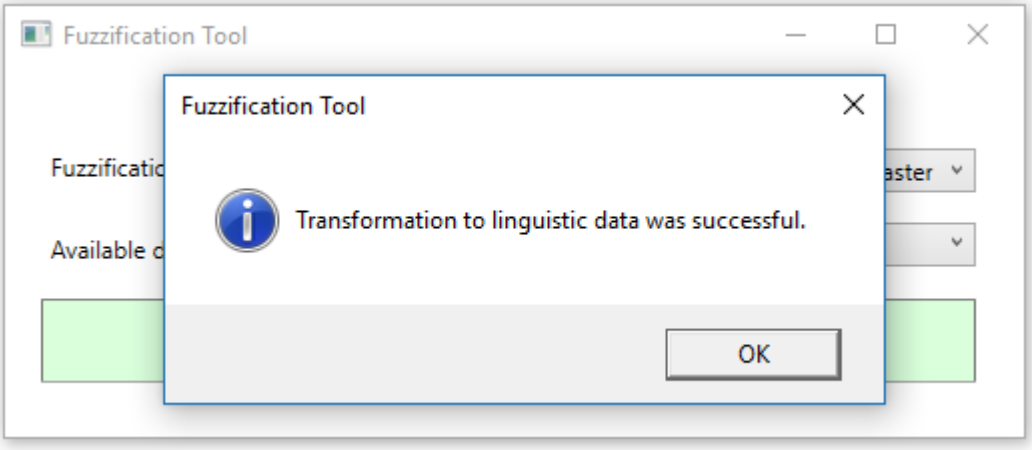
Obr. 3.4: Diagram tried potomkov Datasetu



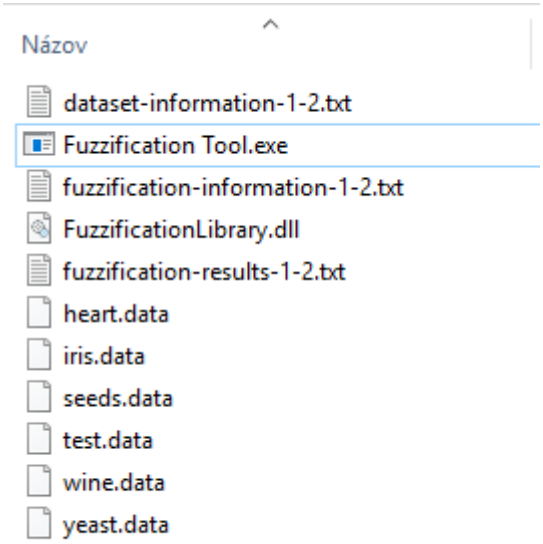
Obr. 3.5: Vstupné menu konzolovej aplikácie



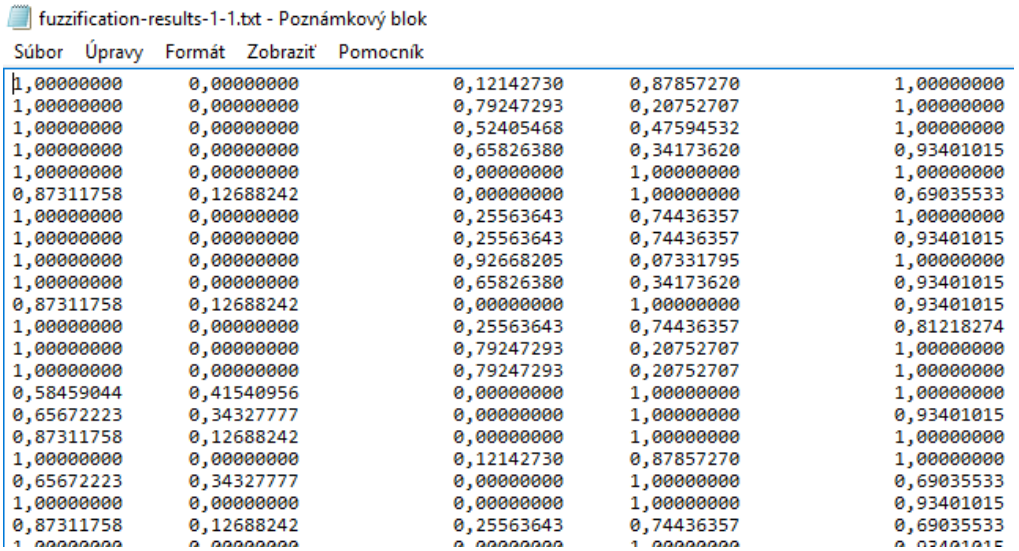
Obr. 3.6: Vstupné menu grafického rozhrania



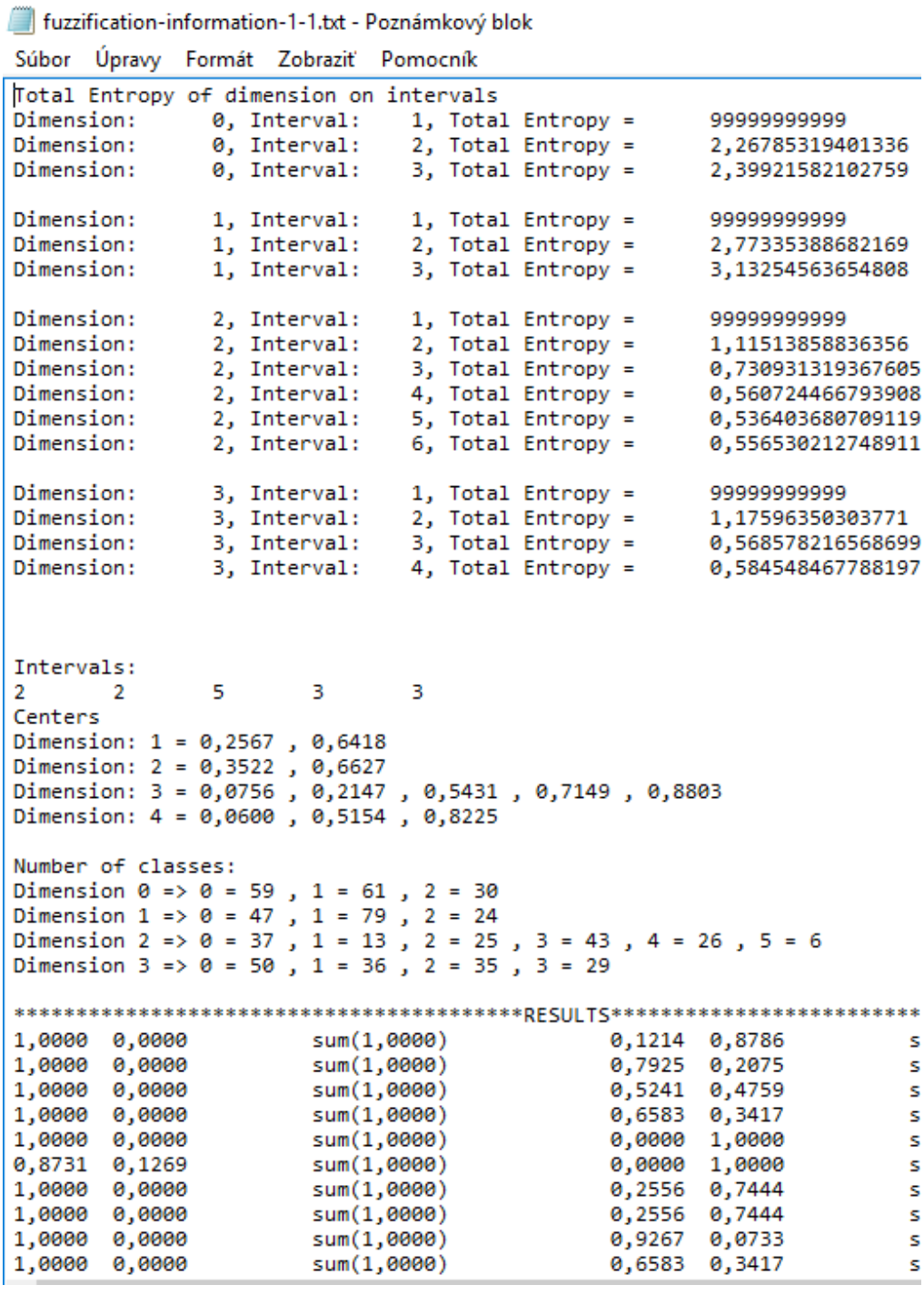
Obr. 3.7: Grafické rozhranie po úspešnom fuzzifikovaní dát



Obr. 3.8: Príklad štruktúry priečinku po fuzzifikácií



Obr. 3.9: Príklad obsahu súboru s výsledkami



Obr. 3.10: Príklad obsahu súboru s podrobnejšími výsledkami

Kapitola 4

Experimentálny výskum

V tejto kapitole opisujem experimentálne porovnanie algoritmov na rôznych vstupných dátach.

4.1 Vstupné súbory

Zdroje dát boli na realizáciu experimentálneho výskumu boli použité dáta z UMCI Machine Learning Repository [38]. Tieto dáta sú využívané na experimentálne porovnania metód a algoritmov získania znalostí. Pre realizáciu experimentov boli použité súbory, ktorých výstupný atribút nadobúda hodnoty z konečnej množiny celočíselných hodnôt.

V experimentov som použila nasledovné súbory, ktorých súbor dát obsahuje:

- **Iris** - výsledky merania parametrov kvetov kosatcov. Výstupný atribút je typ jedného z troch druhov kvetov.
- **Heart** - výsledky merania rôznych ukazovateľov pacientov, trpiacich na srdcovocievne ochorenie. Výsledný atribút opisuje stupeň vážnosti ochorenia.
- **Seeds** - dáta získané ako výsledok merania geometrických vlastností zŕn troch rozličných odrôd pšenice v Inštitúte geofyziky Poľskej akadémie vied.
- **Wine** - dáta chemických ukazovateľov rôznych druhov juhotalianských vín. Vstupné

atribúty obsahujú ingrediencie vo víne. Výstupný atribút určuje jeden z troch druhov vína.

- **Yeast** - slúži na predikciu umiestnenia proteínov v bunkách kvasiniek.

Základné charakteristiky vybraných súborov sú:

- počet pozorovaní,
- počet vstupných atribútov, nadobúdajúcich lingvistické alebo číselné hodnoty,
- počet variantov výstupného atribútu,
- hodnota počiatočnej chyby výberu.

V tabuľke č. 4.1 sú hodnoty charakteristík vybraných súborov dát.

Súbor dát	Počet pozorovaní	Počet výstupných atribútov	Počet lingvistických atribútov	Počet číselných atribútov	P. variantov výstupného atribútov	Počiatočná chyba výberu
Heart	270	13	8	5	2	0,4444
Iris	150	4	0	4	3	0,6667
Seeds	210	7	0	7	3	0,3490
Wine	178	13	0	13	3	0,6011
Yeast	1484	8	2	6	10	0,6880

Tabuľka 4.1: Hodnoty charakteristík vybraných súborov dát.

4.2 Súbor údajov Iris

4.2.1 Opis vstupných dát

Je to najznámejšia databáza, ktorá sa nachádza v literatúre rozpoznávania vzorov. Súbor údajov obsahuje 3 triedy po 50 prípadoch, kde každá trieda sa vzťahuje na typ rastliny

dúhovky. Jedna trieda je lineárne oddeliteľná od ostatných 2; Tieto nie sú lineárne oddeliteľné od seba.

Prvé atribút je dĺžka sepálu v cm, druhý atribút je šírka sepálu v cm, tretí atribút dĺžka okvetných lístkov v cm, štvrtý atribút je šírka okvetných lístkov v cm. Výstupný atribút má triedy - Iris Setosa, Iris Versicolour, Iris Virginica.

Atribút	**Min**	**Max**	**Mean**	**SD**	**Korelácia**
Dĺžka sepálu	4,3	7,9	5,84	0,83	0,7826
Šírka sepálu	2,0	4,4	3,05	0,43	-0,4194
Dĺžka okvetného lístka	1,0	6.9	3.76	1.76	0.9490
Šírka okvetného lístka	0.1	2.5	1.20	0.76	0.9565

Tabuľka 4.2: Súhrnné štatistiky pre databázu Iris

4.2.2 Výsledky fuzzifikácie

Výsledky fuzzifikácie sú v tabuľke č.4.2.2. V tabuľke som vybrala som hodnotu entropiu pre atribút prvý a umiestnenie centier pre atribút štvrtý.

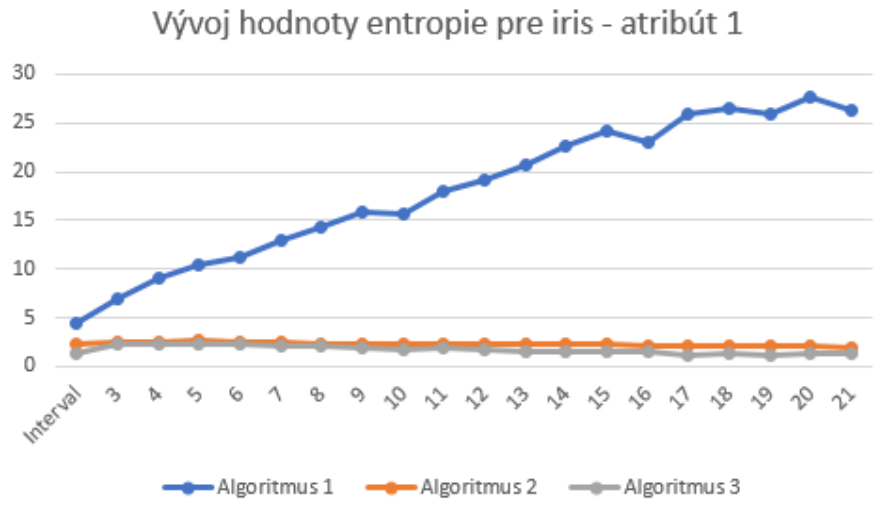
4.2.3 Experiment - Vývoj hodnoty entropie

Experiment spočíva v sledovaní vývoja hodnoty entropie prvého atribútu v intervale $< 2,22 >$. Hodnota entropie pri Algoritme 1 stúpa, zatiaľ čo algoritmy čo používajú

	Počet intervalov	**Hodnota entropie pre atribút 1**	**Umiestnenie centier pre atribút 4**
Algoritmus 1	2,2,3,3,3	4,4680315	0,0600 0,5154 0,8225
Algoritmus 2	2,2,5,3,3	2,2678531	0,0600 0,5154 0,8225
Algoritmus 3	3,3,4,4,3	6,8543783	0,0600 0,4722 0,6595 0,8822
Algoritmus 4	5,2,3,4,3	1,6914607	0,0000 0,0417 0,3750 0,4583

Tabuľka 4.3: Výsledky fuzzifikácie Iris databázy.

váženú entropiu, sa hodnoto udržujú na rovnakej úrovni. Počet tried na intervale tri je priemerne pre triedy 60, 60, 30.



Obr. 4.1: Vývoj hodnoty entropie pre iris - atribút 1.

4.3 Súbor údajov vína

4.3.1 Opis vstupných dát

Tieto údaje sú výsledkom chemickej analýzy vín pestovaných v rovnakom regióne v Taliansku, ale odvodených od troch rôznych kultivarov. Analýza určila množstvo 13 zložiek, ktoré sa nachádzajú v každom z troch druhov vín.

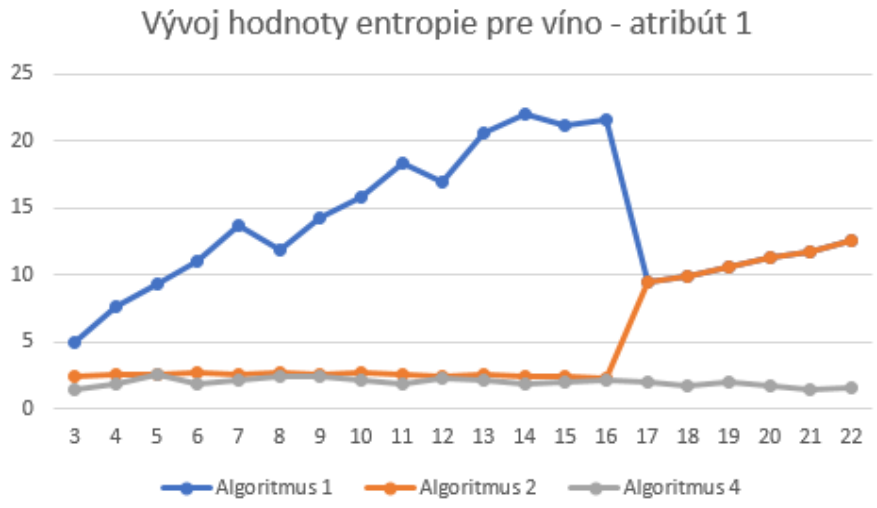
Počet inštancií je 178, z toho v triedach je 59, 71, 48 položiek. Prvý atribút je identifikátor triedy. Počet atribútov má 4 číselných, 1 prediktívny atribút a triedu. Názvy atribútov: Alkohol , Kyselina jablčná, Popol , Alkalinity popola , Horčík , Celkové fenoly , Flavanoidy , Neflavanoidné fenoly , Proantokyaníny , Intenzita farieb , Odtieň , OD280 / OD315 zriedených vín , Prolín.

	Hodnota entropie pre atribút 1	Hodnota entropie pre atribút 1 $(I+1)$	Umiestnenie centier pre atribút 1
Algoritmus 1	4,8846493	7,6232103	0,3289 0,6919
Algoritmus 2	2,7733194	2,8339819	0,3289 0,6919
Algoritmus 3	4,8846493	7,6232103	0,2779 0,5232 0,7550
Algoritmus 4	1,1502002	2,1532059	0,1395 0,1553

Tabuľka 4.4: Výsledky fuzzifikácie Wine databázy.

4.3.2 Výsledky fuzzifikácie a experimenty

Výsledky fuzzifikácie sú v tabuľke č.4.3.2. Počet tried pre atribút prvý je 61, 56, 61. Experiment spočíva v sledovaní vývoja hodnoty entropie prvého atribútu.



Obr. 4.2: Vývoj hodnoty entropie pre víno - atribút 1

4.4 Súbor údajov kvasníc (Yeast)

Údaje sú na predpovedanie bunkovej lokalizácie proteínov. Počet inštancií je 1484 pre súbor údajov o kvasinkách. Počet atribútov pre súbor kvasiniek je 9 (8 prediktívnych, 1 názov):

1. Názov série: Prístupové číslo pre databázu SWISS-PROT
2. mcg: McGeochova metóda na rozpoznávanie signálových sekvencií.
3. gvh: von Heijneov metóda pre rozpoznávanie signálnej sekvencie.
4. alm: Skóre prognózového programu ALOM membránovej oblasti.
5. mit: Skóre analýzy obsahu aminokyselín.
6. erl: Prítomnosť substringu "HDEL". Binárny atribút.
7. pox: Peroxizomálny cieľový signál na C-konci.
8. vac: Skóre rozlišovacej analýzy obsahu aminokyselín v Vakuolárne a extracelulárne proteíny.
9. nuc: Skóre analýzy jadrových lokalizačných signálov.

Distribúcia tried je nasledovná :

- CYT (cytosolický alebo cytoskeletálny) - 463
- NUC (nukleárne) - 429
- MIT (mitochondriálna) - 244
- ME3 (membránový proteín, žiadny N-koncový signál) - 163
- ME2 (membránový proteín, neštiepený signál) - 51
- ME1 (membránový proteín, štiepený signál) - 44
- EXC (extracelulárne) - 37
- VAC (vakuolárne) - 30
- POX (peroxisomálny) - 20
- ERL (endoplazmatický retikulový lumen) - 5

Výsledky fuzzifikácie sú v tabuľke č. 4.5

4.5 Súbor údajov Statlog (srdce)

Táto sada dát je databáza srdcových ochorení. Počet inštancií je 270 a atribútov je 13. Predpokladaná premenná je absencia (1) alebo prítomnosť (2) ochorenia srdca. Typy atribútov:

- Numerické: 1,4,5,8,10,12.
- Usporiadané: 11.
- Binárne: 2,6,9.
- Nominálna hodnota: 7,3,13.

V tabuľke č. 4.6 sú výsledky fuzzifikácie dát.

4.6 Súbor dát semien

Súbor dát je z merania geometrických vlastností jadier, ktoré patria do troch rôznych odrôd pšenice. Mäkká röntgenová technika a balík GRAINS boli použité na zostavenie všetkých siedmich atribútov s reálnou hodnotou.

Informácie o atribútoch: Na zostrojenie údajov sa meralo sedem geometrických parametrov pšeničných jadier: oblasť A, obvod P, kompaktnosť C = $4 * pi * A/P^2$, dĺžka jadra, šírka jadra, koeficient asymetrie, dĺžka drážky jadra.

V nasledujúcej tabuľke 4.7 sú výsledky pre súbor dát semien.

4.7 Zhrnutie výsledkov

Algoritmus 4 efektívne zoradí centra. Pri Algoritme 1 hodnota entropie stúpa smerom nahor. Algoritmus 2 pomocou váženej entropie je dobrý na určenie počtu intervalov, ak triedy majú nerovnomerne rozdelené prvky. Algoritmus 3 dáva vyšší počet intervalov kvôli tomu, že prahová hodnota bola nastavená nízko.

	Hodnota entropie pre atribút 1	**Hodnota entropie pre atribút 1 - I+1**	**Umiestnenie centier pre atribút 1**
Algoritmus 1	4,8846493482439	7,6232103794469	0,3289 0,6919
Algoritmus 2	2,77331942036114	2,83398191261714	0,3289 0,6919
Algoritmus 3	4,8846493482439	7,6232103794469	0,2779 0,5232 0,7550
Algoritmus 4	1,15020020568059	2,15320594479199	0,1395 0,1553

Tabuľka 4.5: Výsledky fuzzifikácie pre súbor dát kvasníc

	Hodnota entropie pre atribút 1	**Hodnota entropie pre atribút 1 - I+1**	**Umiestnenie centier pre atribút 1**
Algoritmus 1	4,8846493482439	7,6232103794469	0,3289 0,6919
Algoritmus 2	2,77331942036114	2,83398191261714	0,3289 0,6919
Algoritmus 3	4,8846493482439	7,6232103794469	0,2779 0,5232 0,7550
Algoritmus 4	1,15020020568059	2,15320594479199	0,1395 0,1553

Tabuľka 4.6: Výsledky fuzzifikácie pre súbor dát srdce

	Hodnota entropie pre atribút 1	**Hodnota entropie pre atribút 1**	**Umiestnenie centier pre atribút 1**
Algoritmus 1	4,8846493482439	7,6232103794469	0,3289 0,6919
Algoritmus 2	2,77331942036114	2,83398191261714	0,3289 0,6919
Algoritmus 3	4,8846493482439	7,6232103794469	0,2779 0,5232 0,7550
Algoritmus 4	1,15020020568059	2,15320594479199	0,1395 0,1553

Tabuľka 4.7: Výsledky fuzzifikácie pre súbor dát - Semená

Záver

Počas tvorby diplomovej práce som sa naučila lepšie chápať algoritmom a ako ich naimplementovať. Nástroj, ktorý som vytvorila by sa mohol v budúcnosti rozšíriť o viaceré súbory dát, ako aj rôzne typy algoritmov. Možno by bolo pekné, bolo súčasťou nástroja vizualizácia dát.

Hlavným cieľom práce bolo implementovať existujúce algoritmy na transformovanie numerických dát na lingvistické dáta a porovnať ich. Úspešne som vytvorila nástroj na fuzzifikovanie dát a porovnala ich.

Literatúra

[1] LEVASHENKO V. - ZAITSEVA E. - KOVALÍK Š., *Projektovanie systémov pre podporu rozhodovania na základe neurčitých dát.* Žilinská univerzita v Žiline/EDIS, 2013. ISBN 978-80-554-0680-0.

[2] Zadeh L., *Fuzzy sets.* Information and Control, vol.8, 1965, pp. 338-353.

[3] Kaufmann A., Gupta M., *Induction to fuzzy arithmetic: theory and applications.* New York : Van Nostrand Reinold Co., 1985, 361 p.

[4] Klir G., Yuan B., *Fuzzy Sets and Fuzzy Logic. Theory and Applications.* Prentice Hall, 1995, 591 p.

[5] Navara M., *Computation with fuzzy quantities.* Proc. of the 7th Conf. of the European Society for Fuzzy Logic and Technology (EUSFLAT), Aix-les-Bains, France, 2011, pp. 209-214.

[6] GREGOR M., *Umelá inteligencia 1* , CEIT, 2014, ISBN 978-80-971684-1-4.

[7] SPALEK.J - JANOTA. A - BLAŽOVIČOVÁ, M. - PŘIBYL, P. *Rozhodovanie a riadenie s podporou umelej inteligencie.* Žilinská univerzita v Žiline/EDIS, 2005. ISBN 80-8070-354-X.

[8] NICKLES. M - SOTTARA, D. *Approaches to Uncertain or Imprecise Rules - A Survey.* In Rule Interchange and Applications, vol. 5858 of *Lecture Notes in Computer Science*, pp. 323-336. Springer, 2009. ISBN 9783642049842.

[9] ROSS, T.J. *Fuzzy Logic with Engineering Applications.*. John Wiley & Sons, 2004, second edition ed. ISBN 0-470-86075-8.

[10] PASISNO, K. M. - YURKOVICH, S. *Fuzzy control*, vol.42. Addison Wesley Longman, 1998. ISBN 0-201-18074-X.

[11] Catlett J., *On Changing Continuous Attributes into Ordered Discrete Attributes.* Lecture Notes on Computer Science, Berlin: Springer-Verlag, vol. 482, 1991, pp. 164-177.

[12] Liu H., Hussain F., Lim Tan C., Dash M., *Discretization: An Enabling Technique.* Data Mining and Knowledge Discovery, Kluwer Academic Publishers, vol.6, 2002, pp. 293-423.

[13] B. Kosko, *Fuzzy entropy and conditioning*, Inform. Sci., vol. 40, pp. 165-174, Dec. 1986.

[14] A. Renyi, *On the measure of entropy and information*, in Proc. Fourth Berkeley Symp. Math. Statistics Probabilit, vol. 1, Berkeley, CA, 1961, pp. 541-561.

[15] R. E. Belahut, *Principles and Practice of Information Theory*, Reading, MA: Addison-Wesley, 1987.

[16] J.Y. Ching et al., *Class-dependent discretization for inductive learning form continuous and mixed-mode data*, IEEE Trans. Pattern Anal. Machine Intell., vol. 17, pp. 641-651, July 1995.

[17] Autor: Sedláková, V., Školiteľ: Girovský P., Oponent: Timko J. *Fuzzy riadenie synchrónneho motora*, Technická univerzita v Košiciach, Rok odovzdania 2013, Počet strán 113s.

[18] García S., Luengo J. Sáes J.A., López V., Herrera F., *A Survey of Discretization Techniques: Taxonomy and Empirical Analysis in Supervised Learning* IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 25, NO. 4, 2013, pp. 734-750.

[19] Liu L., Wong A., Wang Y., *A global optimal algorithm for class-dependent discretization of continuous data.* Intelligent Data Analysis, vol. 8, 2004, pp. 151-170.

[20] Singh G.P., Singh B., *Simulink Library Development and Implementation for VLSI Testing in Matlab*, Communications in Computer and Information Science, vol.169 CCIS, 2011, pp. 233-240.

[21] Bakar A., Othman Z., Shuib N., *Building a New Taxonomy for Data Discretization Techniques*, Proc. on Int. Conf. on Data Mining and Optimization (DMO), 2009, pp. 132-140.

[22] Yang Y., Webb G. I., Wu X., *Discretization methods*, Data Mining and Knowledge Discovery Handbook, 2010, pp. 101-116.

[23] Garcia M., Lucas J. P., Batista V. F. L., *Multivariate discretization for associative classification in a sparse data application domain*, Proc. of the 5^th Int. Conf. on Hybrid Artificial Intelligent Systems (HAIS), 2010, pp. 104-111.

[24] Chelbus B., Nguyen S.H., *On Finding Optimal Discretizations for Two Attributes*, Lecture Notes in Artificial Intelligence, vol.1424, 1998, pp. 537-544.

[25] Wong A.K., Chiu D.K., *Synthesizing Statical Knowledge from Incomplete Mixed-Mode Data*. IEE Trans. on Pattern Analysiss and Machine Intelligence, vol.9, no.6, 1987, pp. 796-805.

[26] Kerber R., *ChiMerge: Discretitation of numeric attributes*. Proc. of the 9th National Conf. on Artifical Intelligence American Association for Artifical Intelligence, 1992, pp.123-128.

[27] Quinlan J.R, *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publ. Inc, San Manteo, California, 1993.

[28] Fayyad U.M., Irani K.B., *Multi-Interval Discretitation of Continuous-Valued Attributes for Classification Learning*, Proc. of the 13th Int. Joint Conf. on Artifical Intelligence, 1993, pp. 1022-1027.

[29] Ventura D., Martinez T. R., *BRACE: A paradigm for the discretization of Continuously valued data.* Proc. of the 7th Annual Florida AI Research Symposium (FLAIRS), 1994, pp. 117-121.

[30] Pazzani M.J., *An iterative improvement approach for the discretization of numeric attributes in bayesian classifiers*. Proc. of the Int. Conf. on Knowledge Discovery and Data Mining (KDD), 1995, pp. 228-233.

[31] Ching J.Y., Wong A.K.C., Chan, K.C.C., *Class-Dependent Diskretization for Inductive Learning from Continuous and Mixed-Mode Data*. IEEE Trans. on Pattern Analysis and Machine Intelligence, vol.17, no.7, 1995, pp. 641-651.

[32] Lee H-M., Chen C.-M., Chen J.-M., Jou Y.-L., *An Efficient Fuzzy Classifier with Feature Selection Based on Fuzzy Entropy.* IEEE Trans. on Systems, Man and Cybernetics, Part B: Cybernetics, vol. 31, no.3, 2001, pp. 426-432. Dostupné online: `http://dx.doi.org/10.1109/3477.931536`

[33] Z. Chi and H. Yan, *Feature evaluation and selection based on an entropy measure with data clustering*, Opt. Eng., vol. 34, pp. 3514-3519, Dec. 1995.

[34] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, New York: Wiley, 1973.

[35] Lin C.J., Lee C.Y., Hong S.J., *An Efficient Fuzzy Classifier Based on Hierarchical Fuzzy Entropy*, International Journal of Information Technology, Vol. 12, No.6, 2006.

[36] Bezdec, J.C., *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.

[37] Fuzzy Logic Toolbox User's Guide. Dostupné online: `https://www.mathworks.com/help/pdf_doc/fuzzy/fuzzy.pdf`

[38] Bache K., Lichman M., *UCI Machine Learning Repository*, Irvine, CA: University of California, School of Information and Computer Science, 2013. Dostupné online: `http://archive.ics.uci.edu/ml`