Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Bakalářská práce

Klasifikace pomocí neuronových sítí a vektorových strojů

Tomáš Dohnal

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Tomáš Dohnal

Informatika

Název práce

Klasifikace pomocí neuronových sítí a vektorových strojů

Název anglicky

Classification with neural networks and vector machines

Cíle práce

Porovnání efektivity klasifikačních algoritmů založených na neuronových sítích a algoritmů založených na vektorových strojích.

Metodika

V práci bude porovnána efektivita klasifikačních algoritmů založených na neuronových sítích a klasifikačních algoritmů založených na vektorových strojích (support vektor machines). Diplomant se zaměří především na případ, kdy klasifikované objekty jsou popsány mnoha parametry a k dispozici jsou pouze relativně malé učební soubory. Pro práci použije open source software přístupný na webu a data z veřejného úložiště dat Kalifornské univerzity, která jsou určena pro testování rozhodovacích modelů.

Doporučený rozsah práce

35-40

Klíčová slova

neuronové sítě, vektorové stroje

Doporučené zdroje informací

BISHOP, C M. *Pattern recognition and machine learning.* [New York]: Springer, 2006. ISBN 0-387-31073-8.

Předběžný termín obhajoby

2018/19 ZS – PEF (únor 2019)

Vedoucí práce

doc. Ing. Arnošt Veselý, CSc.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 23. 2. 2018

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 23. 2. 2018

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 25. 02. 2019

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci " Klasifikace pomocí neuronových sítí a vektorových strojů" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne _____

Poděkování

Rád bych touto cestou poděkoval mé rodině, za podporu v situacích studijních i mimo ně. Dále pak doc. Ing. Arnoštovi Veselému, CSc. za podporu při řešení mé bakalářské práce.

Klasifikace pomocí neuronových sítí a vektorových strojů

Abstrakt

Cíle práce je porovnání efektivity klasifikačních algoritmů založených na neuronových sítích a algoritmů založených na vektorových strojích.

V práci bude porovnána efektivita klasifikačních algoritmů založených na neuronových sítích (neural net) a klasifikačních algoritmů založených na podpůrných vektorových strojích (support vektor machines). Bakalant se zaměří především na případ, kdy klasifikované objekty jsou popsány mnoha parametry a k dispozici jsou pouze relativně malé učební soubory. Pro práci použije open source software přístupný na webu a data z veřejného úložiště dat Kalifornské univerzity, která jsou určena pro testování rozhodovacích modelů.

Klíčová slova: neuronové sítě, podpůrné vektorové stroje, klasifikace, Rapidminer, UCI

Classification with neural networks and vector machines

Abstract

The thesis deals with the research of classification algorithms based on neural nets and support vector machines and their comparison on exact dataset example.

The aim is to focus on example, where subjects of classification are described with a lot of parameters and there is only limited, relatively small number of learning folders. For research is used open source software, accessible on web and example dataset could be found in open database from Caltech University, wchich are purposed for this kind of testing decision-making processes.

Keywords: neural network, support vector machines, classification, Rapidminer. UCI

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Základním projevem živé hmoty, je schopnost přizpůsobit se okolnímu prostředí neboli „adaptace“, proces přizpůsobení je pak „adaptivní proces“.

Život je adaptivní proces, snaha přežít. Při vyčerpání adaptivních schopností život zaniká. Při adaptaci dochází ke ztrátě materiální, energické, nebo informativní. Opakování adaptace na určitou změnu, má u živých organismů za důsledek minimalizování těchto ztrát. Tudíž tento proces opakování je zdrojem zkušeností, které jsou organismem zhodnocovány a slouží tak ke snížení ztrát za účelem adaptace. Dochází tedy k „učení". To má v realitě mnoho podob, odlišností a náročností, nelze je zobecnit. Můžeme ale vytvořit přibližný obraz o tom, jak probíhají procesy učení alespoň v nižších formách a aplikovat je na výpočetní techniky, a tím je zefektivnit. K tomu napomáhá využití metod tzv. umělé inteligence, která otevírá zcela nové možnosti strojového zpracování informací. Součástí zpracovávání informací je metoda rozpoznávání neboli klasifikace, slouží k zařazení objektů, jevů a situací do tříd i k analýze scén. To jsou základní metody pro rozpoznávání obrazové informace, počítačového vidění, rozpoznávání akustických signálů včetně řečových až po porozumění přirozené řeči.

Touto bakalářskou prací bych chtěl porovnat klasifikační metody a zjistit jejich efektivnost v určitých případech.

1.1 Cíle práce:

Porovnání efektivity klasifikačních algoritmů založených na neuronových sítích a algoritmů založených na vektorových strojích.

1.2 Metodika:

V práci bude porovnána efektivita klasifikačních algoritmů založených na neuronových sítích a klasifikačních algoritmů založených na vektorových strojích (support vektor machines). Zaměření se především na případ, kdy klasifikované objekty jsou popsány mnoha parametry a k dispozici jsou pouze relativně malé učební soubory. Pro práci bude použit open source software „RapidMiner“ a data z veřejného úložiště dat Kalifornské univerzity (UCI), která jsou určena pro testování rozhodovacích modelů. Tato data budou předzpracována a připravena pro aplikování v programu. RapidMiner bude použit pro nalezení optimálního nastavení jednotlivých parametrů k oběma modelům, v závislosti na konkrétních datech a úloze. Dále budou klasifikační algoritmy s optimálními parametry pozorovány a jejich výsledky srovnávány mezi sebou navzájem v programu RapidMiner. Datový set, který máme k dispozici osahuje informace již známých výsledků. Proto bude známý soubor rozdělen na trénovací, validační a testovací množinu. Na základě známých výsledků a hodnotách jejich atributů se pokusíme naučit náš vytvořený model pro určení třídního atributu na nových případech, kde výslednou klasifikaci model neuvidí a budeme zkoumat jejich úspěšnost. Klasifikátory neuronových sítí a podpůrných vektorových strojů budou trénovány na trénovacích datech, dále bude ověřeno jejich naučení na validačních datech a testováno na testovacích datech. K výsledku bude nastavené prahové hledisko pozorovatelem úměrně k dané problematice. Výsledkem práce bude odůvodněné, grafické představení efektivnějšího modelu vzhledem k zadanému datovému souboru a vzhledem k požadovanému předmětu sledování.

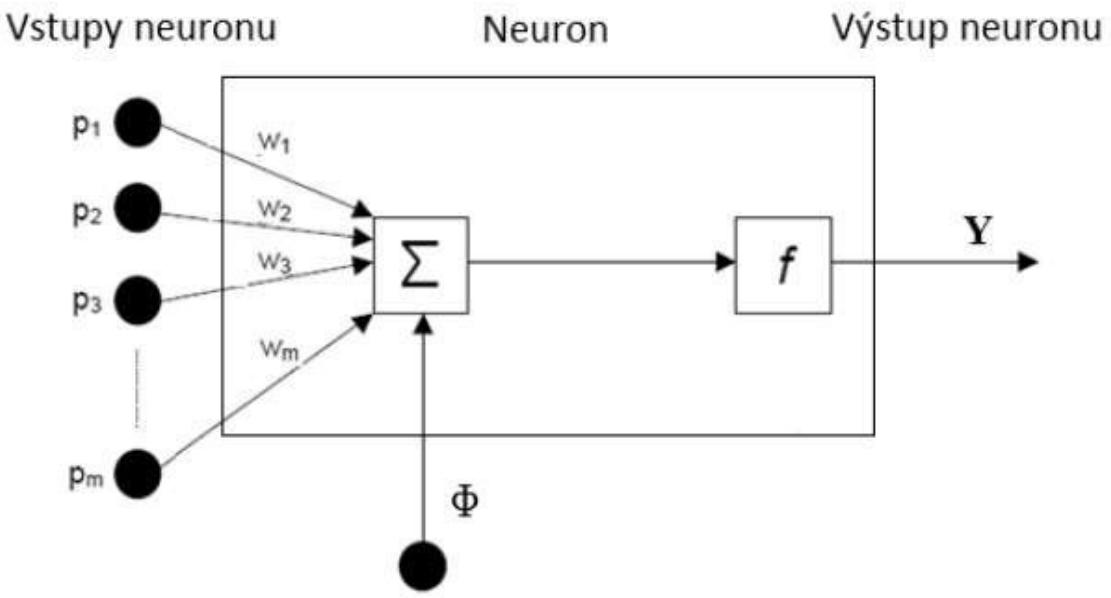
2 Teoretická východiska

2.1 Neuronové sítě (NN)

Současná doba je charakteristická snahou o automatizování složitých úloh, které jsou náročné především na rozpoznávání nejrůznějších objektů. Dynamický vývoj v oblasti zvyšování výpočetního výkonu počítačů a vývoj senzorických systémů klade zvýšené nároky na tzv. real–time aplikace, tj. aplikace probíhající v reálném čase, kde jedním z klíčových kritérií je čas. U těchto aplikací jsou také požadovány efektivní a rychlé algoritmy pro klasifikaci objektů, kde se stále častěji prosazují metody umělé inteligence, do které patří i umělé neuronové sítě. [2]

Umělé neuronové sítě se skládají z jednotlivých neuronů a jsou strojovou obdobou biologické předlohy lidské nervové soustavy, a to nejen jejich stavbou, ale i procesem řešení úloh.

Neuronové sítě jsou soustavou paralelně zapojených perceptronů (každý perceptron představuje neuron). Jednoduchý perceptron je vyobrazen na Obr. 5. Na vstupy sítě přichází informace, která je dále dělena na dílčí segmenty. To znamená rozdělení vstupní informace neboli „vektoru příznaků“ na části. Kdy každý neuron vyhodnocuje přidělený segment na základě předchozích znalostí, zadaných a parametrů sítě. Testování probíhá sekvenčně. Po zpracování je segment poslán výstupem k dalšímu zhodnocení na následující neuron v další paralelní vrstvě sítě a proces je opakován, dokud segment takto neprojde celou neuronovou sítí až na konečný výstup neuronové sítě.
Vstupy perceptronu p_i mohou mít přiděleny váhové koeficienty $w_{i.}$ a tím měnit konkrétní zpracování příchozí informace. Vstup každého neuronu je pak násoben jeho váženou hodnotou w_i. Všechny váhové vstupy do neuronu jsou následně sečteny ve vnitřním bloku Σ a porovnávány s nastavenou prahovou hodnotou φ, v případě, že je součet vah vyšší, než prahová hodnota φ, je pak signál připuštěn na výstup. Jinými slovy, pokud ve vztahu prahová hodnota a suma vah platí nerovnost, výstupní funkce je aktivována.



Obrázek 1 : Schéma perceptronu (neuronu). [6].

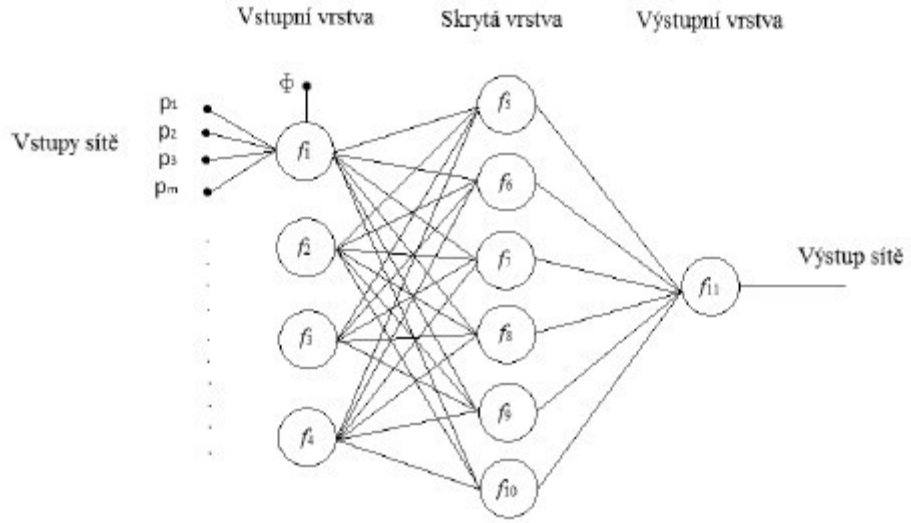
Asi největší předností neuronových sítí je schopnost učení, abstrakce a generalizace. Nezávislost na statistickém rozložení a vysoká tolerance šumu v datech je také oproti ostatním metodám výhodou. To vše je ale vykoupeno složitým systémem nastavení sítě a jejích parametrů a dlouhou dobou učení.[1]

U neuronových sítích se nejedná pouze o jeden klasifikátor, ale o skupinu klasifikačních algoritmů se stejnou typologii.[10]. V průběhu trénovacího procesu, kdy se síť učí na trénovací sadě, se trénovací algoritmus adaptuje. A to tak, že v závislosti na výsledcích přizpůsobuje hodnoty váhovacích koeficientů w_i na vstupních perceptronech. Tím je zajištěna změna na výstupu neuronu a tím i na výstupu celé sítě v jednotlivých cyklech učení. Výstupy sítě jsou porovnávány s požadovaným výstupem, tedy učení s učitelem (supervised learning). Toto učení a jeho adaptivní změny vah jsou předem definovány strukturou sítě, která se již v průběhu procesu nemění. Kvalita učení je současně podmíněna adekvátní volbou trénovací sady dat. Požadovaný je výběr takové trénovací sady, která zajistí maximalizování diskriminační schopnosti při minimalizaci objemu dat. Vhodně navržená a natrénovaná síť by měla být schopna na principu naučeného modelu vyhodnocovat testovací data, tedy nová data, se kterými se síť prozatím nesetkala.

Správné natrénovaní sítě spočívá v nalezení optimální míry naučení z trénovací sady. Natrénovat síť se 100% úspěšností rozpoznání konkrétní testovací sady není vhodné, model pak ztrácí obecnost a schopnost jeho přenesení na nová data a model nepodává optimálními výsledky. Tento nežádoucí stav se nazývá přeučení sítě (overfitting).

Neurony, které nejsou výstupy či vstupy celé sítě, ale jejich výstupy a vstupy slouží jako mezičlánek zpracování uvnitř sítě, se nazývají skrytá vrstva. Skrytých vrstev může síť obsahovat libovolný počet zvolený pozorovatelem při vytváření struktury sítě, hovoříme pak o vícevrstvých neuronových sítích. Princip fungování neuronových sítí je totožný a není závislý na počtu neuronů nebo jejich vrstev.

Bias neuron, je speciální typ neuronu, který má jednotlivě vazbu na každý neuron dané vrstvy. Každá vrstva sítě má jeden bias neuron. Funkce biasu je přidělování váhy koeficientů na jednotlivých vstupech neuronů a jejich aktivační funkci. [9],[11],[12],[13],[23].



Obrázek 2 : Schéma neuronové sítě.[12]

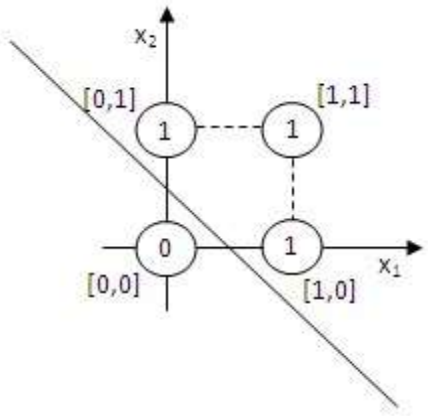
2.1.1 **Učení umělé neuronové sítě**

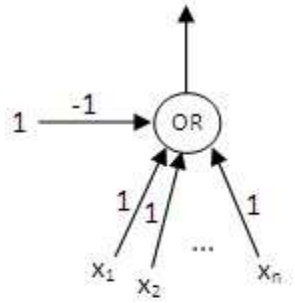
Stavba nervové tkáně u člověka se od narození téměř nemění, učení je dáno nikoliv přestavbou sítě, ale především posílením nebo naopak zeslabením vazeb prostřednictvím synapsí. Časté a intenzivní impulzy mohou synapse mezi neurony změnit trvale. U umělé neuronové sítě proces učení moderuje pomocí nastavení vah vstupů, popř. změnou aktivační funkce. Jakým způsobem jsou tyto změny prováděny, záleží na konkrétní síti a algoritmu učení, který je pro danou síť používán.

2.1.2 **Klasifikační možnosti neuronu a neuronové sítě**

Jednotlivé neurony mají pouze omezenou výpočetní sílu. Již jediný binární neuron však dokáže rozdělit prvky množiny do dvou lineárně separabilních skupin, a to proložením nadroviny v daném n-rozměrném prostoru. Díky jedinému neuronu lze tedy modelovat základní logické funkce jako AND, OR, NOT (obr. 7, obr. 8). Důležitá je zde právě lineární separabilita skupin. Pokud není splněna nelze rozdělit skupiny jedinou nadrovinou.

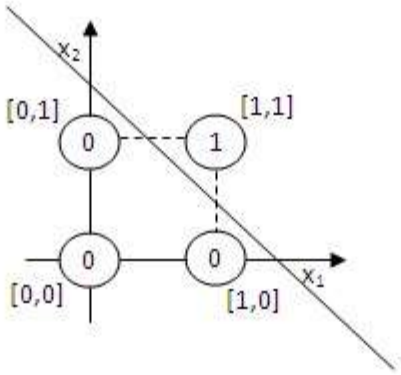
x	y	z
1	1	1
1	0	1
0	1	1
0	0	0

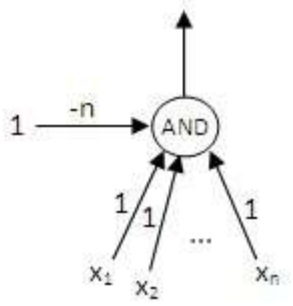




Obrázek 3 : Pravdivostní tabulka pro logickou funkci OR, geometrické znázornění výpočtu OR, neuronová realizace OR

x	y	z
1	1	1
1	0	0
0	1	0
0	0	0

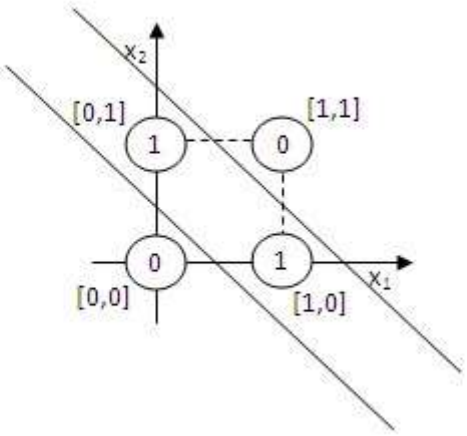


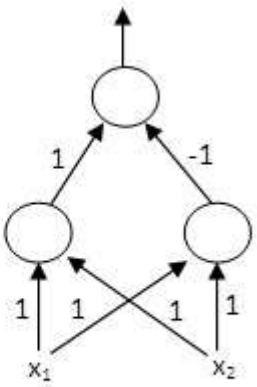


Obrázek 4 : Pravdivostní tabulka pro AND, geometrické znázornění výpočtu AND, neuronová realizace AND

Právě tento problém v separabilitě může nastat u logické funkce XOR. Z rozmístění hodnot (obr. 9) a ze stylu učení je zřejmé, že neuron není schopen dokončit proces učení, bude se stále učit, ale nikdy se nenaučí.

x	y	z
1	1	0
1	0	1
0	1	1
0	0	0





Obrázek 5 : Pravdivostní tabulka pro XOR, geometrické znázornění výpočtu XOR, neuronová

Spojováním jednotlivých neuronů lze však dosáhnout jakékoliv logické funkce, a to i funkce XOR (obr. 5). Problém XOR lze tak vyřešit přidáním tzv. skrytých neuronů, které slouží jako detektory situací. První skrytý neuron slouží pro detekci situace, kdy je aktivován alespoň jeden vstupní neuron. Druhý detekuje situaci, kdy jsou aktivovány oba vstupní neurony. Výstupní neuron je pak aktivován prvním neuronem a omezován druhým neuronem. [7],[13], [14], [27]

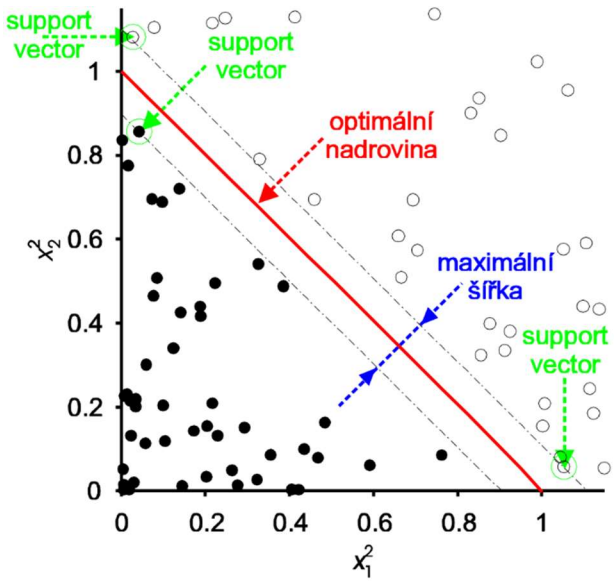
2.2 Podpůrné vektorové stroje (SVM)

Princip metody podpůrných vektorů, je používán v náročnějších, lineárně neseparovatelných úlohách, které se převádí na řešitelné (lineárně separovatelné) úlohy pomocí tzv. jádrové transformace (kernel transformation).

Data jsou převedena na vektory v příznakovém prostoru. Vizuální zobrazení je vidět na obr. (vzhledem k 2D zobrazení jsou vektory reprezentovány jednotlivými body). Podpůrné vektory zajišťují rozdělení tohoto souboru dat do skupin (tříd) na základě podobnosti dat, pomocí oddělovací nadroviny (lineárního klasifikátoru) tak, aby byl oddělovací prostor mezi třídami co možná největší. Tato nadrovina je vypočítána podle příslušné kernelové funkce. Oddělující nadrovina protíná příznakový prostor a rozdělí ho tak na vícedimenzionální příznakový prostor. Každá vzniklá třída reprezentuje jednu dimenzi. Podpůrné vektory (znázorněny body v zeleném kroužku na Obr.6) jsou vektory na okraji třídy, které se nachází nejblíže k oddělovací nadrovině (lineárnímu klasifikátoru) [13].

Podpůrné vektory určují polohu roviny v prostoru. A je jim přiřazena nejvyšší váha při analogii váhování. 8]. Mají tedy větší vliv na klasifikaci než ostatní vektory vektorového prostoru, které jsou od oddělovací nadroviny vzdálenější.

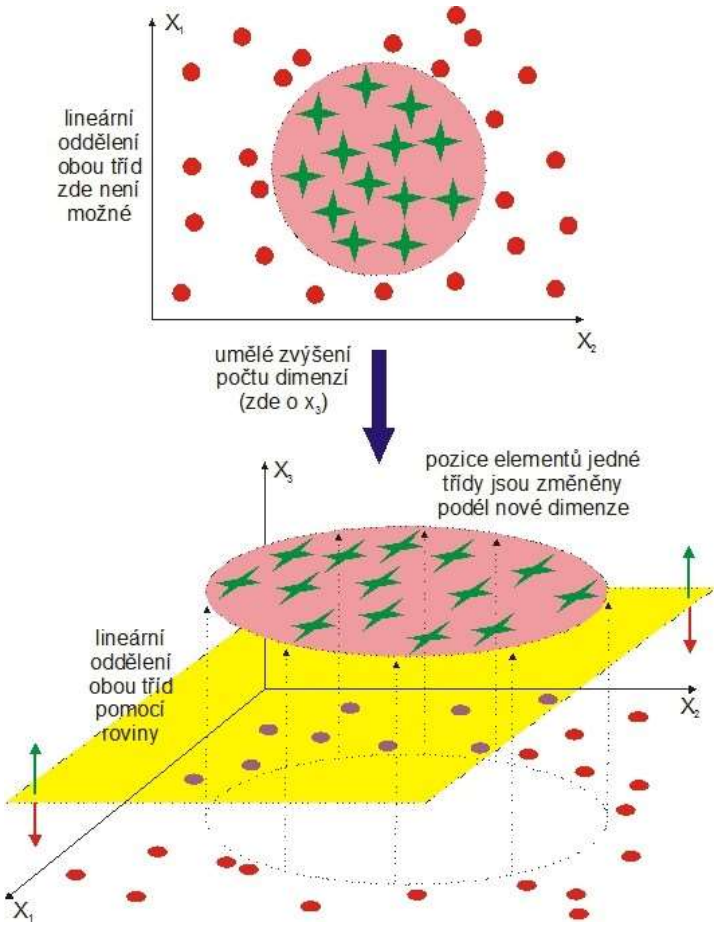
Přerušované přímky na obr. 6 vyznačují okraje oddělovacího prostoru (margin) mezi třídami.



Obrázek 6 : Naznačení umístění podpůrných vektorů. Optimální oddělovací hranice [13]

Podobnost mezi SVM a neuronovými sítěmi je na první pohled zřejmá, avšak zatímco neuronové sítě počítají empirické riziko nacházející se v několika lokálních minimech, SVM si

vystačí s jedním globálním minimem. Účinnost této metody při rozpoznávání je velmi vysoká. [9], [15]



Obrázek 7 : Princip vzniku lineárního oddělení dvou tříd (přidáním jedné dimenze z 2 na 3)

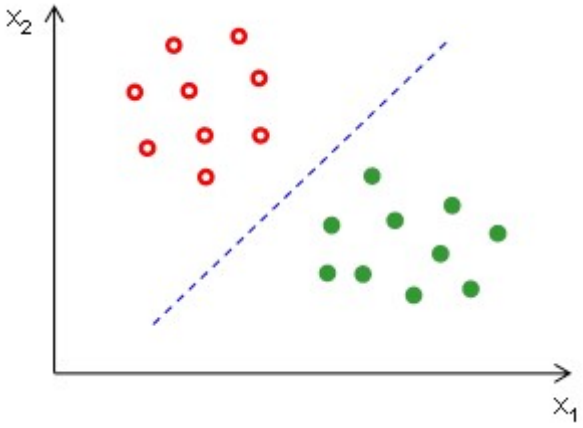
Podpůrné vektory jsou podstatné, protože jejich hlavní funkci je podpora oddělovací nadroviny. Ostatní body nejsou pro oddělení klasifikačních tříd vůbec zapotřebí. Z toho pro nás vyplývá, že tato metoda je schopna najít specifické trénovací příklady, které jsou pro nalezení oddělovače zapotřebí [16]. Tím se i liší od jiných algoritmů, které jsou založeny na trénování všemi příklady, což vede při velkém počtu k vysoké výpočetní náročnosti. Jak z obrázku 7 vyplývá, je těchto podpůrných vektorů většinou mnohem méně než celkových datových bodů, což zaručuje efektivitu.

Metoda SVM je zařazována mezi takzvané jádrové funkce (kernel machines). Toto zařazení vyplývá ze snahy o zjištění jádrové funkce, která nám zajistí, že pokud ji aplikujeme na vstupní dvojice dat, nemusíme už počítat úplný seznam atributů pro každý datový bod, k tomu abychom určili polohu lineárního oddělovače ve vícerozměrném prostoru [16]. Tato jádrová funkce nemá

žádný zvláštní tvar, ale pouze odpovídá jednomu z vícerozměrných prostorů. Je dokázáno, že při použití jádrových funkcí budeme schopni nalézt lineární oddělovače klasifikačních tříd v prostorech, které mohou mít velký počet rozměrů. [8],[16], [21], [26]

2.2.1 **Klasifikace pomocí hranic**

Vícerozměrná data můžeme znázornit v prostoru, v němž zobrazené body odpovídají jednotlivým objektům (či subjektům) a jehož dimenzionalita odpovídá počtu proměnných, kterými jsou objekty popsány. Pokud se v datech vyskytují skupiny objektů, které chceme od sebe oddělit, zřejmě nás intuitivně napadne nakreslit hranici, která bude prostor rozdělovat tak, aby byly na jedné straně od hranice objekty z jedné třídy a na druhé straně hranice objekty z druhé třídy (obr. 8).

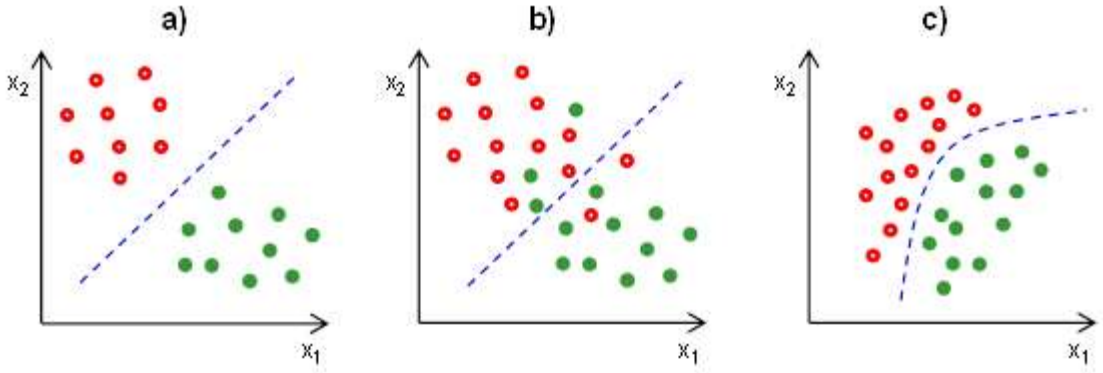


Obrázek 8 : Ilustrace klasifikace pomocí hranice

Hranice jsou tvořeny obecně nadrovinami o rozměru o jednotku menší, než je rozměr prostoru – ve dvourozměrném prostoru je to tedy křivka (ve speciálním lineárním případě přímka), v trojrozměrném prostoru plocha (v lineárním případě rovina), atd. Způsoby určení oddělujících hranic závisí jednak na vlastnostech klasifikačních tříd a jednak na kritériích, která použijeme pro optimalizaci polohy hranic. Co se týče vlastností klasifikačních tříd, zajímá nás zejména:

- zda se jejich obrazy vyskytují v navzájem překrývajících se oblastech, či nikoliv – v tom případě hovoříme o separabilních či neseparabilních skupinách
- zda je možné skupiny objektů oddělit lineární hraniční plochou, či zda je vhodnější použít plochu nelineární.

Na základě kombinací výše uvedených vlastností mohou nastat celkem tři situace, které jsou znázorněny na obr. 9, tedy lineárně separabilní úloha, lineárně neseparabilní úloha s lineárně separovanými třídami a nelineárně separabilní úloha.



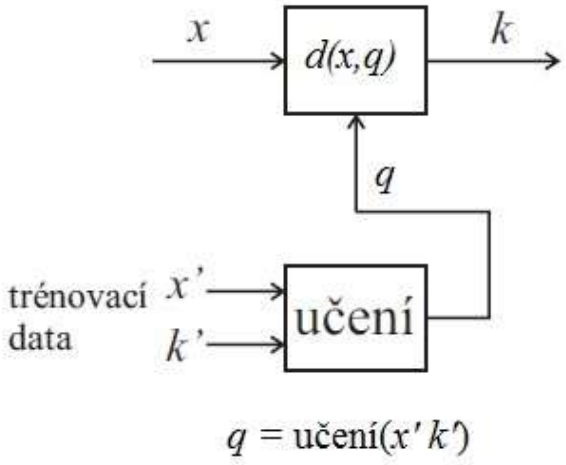
Obrázek 9 : Případy separability klasifikačních tříd-a) lineárně separabilní úloha; b) lineárně neseparabilní úloha, ovšem s lineárně separovanými třídami; c) nelineárně separabilní klasifikační úloha.

[1] [17] [18] [19] [20] [22]

2.3 Proces trénování

Proces nastavení klasifikátoru nazýváme učením (trénováním). Učení se může odehrávat ve dvou různých variantách: s učitelem, bez učitele.

Učení s učitelem (Obr. 10) je založeno na využití trénovacích dat. Učící funkce se skládá z dvojic, kde je vždy jeden vstup „x“ a jemu odpovídající výstup „k“ (dány učitelem). Klasifikátor je pak pomocí natrénování těchto dat a implementování naučených znalostí schopen odhadnout ze skutečného vstupu x výstup k. Tím, že porovnává svůj nynější výstup již s naučenými výstupy a modifikuje váhy spojení, aby dosáhl co největší shody, je schopen zařadit do tříd jakákoliv, i dříve neviděná, vstupní data.



Obrázek 10 : Schéma učení s učitelem [2]

Případ kdy klasifikátor nemá trénovací data k dispozici, nebo je nevyužívá, se nazývá učení bez učitele. V tomto případě si klasifikátor sám odvozuje správnost ze svých výstupů zpětnou vazbou (samoorganizace). Místo učící množiny jsou zde využívány výstupy z klasifikátoru. Výhody tohoto druhu učení se doceňují zejména, když není klasifikace dat známá nebo je příliš složitá [23]

2.4 RapidMiner

RapidMiner je datová vědecká softwarová platforma, původem z Německa, vyvinutá společností stejného jména, která poskytuje integrované prostředí pro přípravu dat, učící stroje, hluboké učení a prediktivní analýzy. RapidMiner má miliony stažení a přes 400 000 aktivních uživatelů včetně společností, jako například BMW, Intel, Cisco, GE a Samsung. RapidMiner si nárokuje vedoucí postavení na trhu softwaru pro vědu před svými konkurenty, jako jsou SAS a IBM. Používá se pro obchodní a komerční aplikace, jakož i pro výzkum, vzdělávání, školení a vývoj aplikací. Podporuje všechny kroky procesu strojového učení, včetně přípravy dat, vizualizace výsledků, ověření a optimalizace modelu.

2.4.1 **Historie:**

RapidMiner, dříve známý jako YALE (Yet Another Learning Environment), byl vyvinut roku 2001 Ralfem Klinkenbergem, Ingo Mierswou a Simonem Fischerem na oddělení umělé inteligence Technické univerzity v Dortmundu. Od roku 2006 byl vyvíjen společností Rapid-I. Společností, kterou založil ve stejném roce Ingo Mierswa a Ralf Klinkenberg. V roce 2007 byl název softwaru změněn z YALE na RapidMiner. V roce 2013 se společnost přejmenovala z Rapid-I na RapidMiner.

2.4.2 **Produkty:**

- RapidMiner Studio
- Automatický model RapidMiner
- RapidMiner Turbo Prep
- Server RapidMiner
- RapidMiner Radoop

Vývojář

Na vývoji open source RapidMiner se podílí asi 50 vývojářů po celém světě, přičemž většina přispěvatelů jsou zaměstnanci RapidMiner. [24]

3 Databáze

V této kapitole je popsána konkrétní užitá databáze, v druhé části jsou uvedeny obecné informace o UCI repository.

3.1 Zvolená databáze

Pro testování algoritmů byly použity datové soubory, obsahující reálné lékařské záznamy o rakovině prsu z onkologického institutu „University Medical Center“, Lublaň z roku 1990. Poděkování M. Zwitter a M. Skolic za vytvoření a poskytnutí údajů. Které jsou veřejně přístupné z UCI repository na webu.

Vybraný celkový dataset se skládá z 286 instancí, které jsou pospány 9 atributy (věk, menopauza, velikost nádoru, node-caps, deg-malig, poprsí, oblast, irradiat) a 1 třídním atributem, který je rozdělen do dvou tříd (recidiva, bez recidivy), kdy jedna třída třídního atributu má 201 a druhá 85 instancí. V předzpracování byly atributy převedeny pozorovatelem na numerické hodnoty. Třídní atribut nabývá binominálních hodnot. Jednotlivé atributy popisují informace o pacientovi a specifické informace o nemoci. Třídní atribut pak popisuje výsledky známé prognózy. Tedy „recidiva“ nebo „bez recidivy“. Prognóza je pro náš model podstatná a budeme trénovat klasifikační algoritmy na rozeznání tohoto třídního atributu.
Na základě známých výsledků a jejich hodnotách atributů se pokusíme naučit náš vytvořený model pro určení třídního atributu na nových případech, které modelu ještě nebyly poskytnuty, a tudíž se s nimi nesetkal. Proces učení bude probíhat na základě informací z instancí ostatních atributů a ověřován na známém třídním atributu.

3.2 UCI repository

UCI Machine Learning Repository je kolekce databází, teorií domén a generátorů dat, které komunita strojového učení používá pro empirickou analýzu algoritmů strojového učení. Archiv vytvořil jako ftp archiv v roce 1987 David Aha a jeho postgraduální student na UC Irvine. Od té doby ji studenti, pedagogové a výzkumní pracovníci po celém světě široce používali jako primární zdroj souborů strojového učení. Jako náznak dopadu archivu byl citován více než 1000krát, což z něj činí jeden z nejlepších 100 nejcitovanějších „zdrojů“ ve všech informatických oborech. Aktuální verzi stránek navrhli v roce 2007 Arthur Asuncion a David Newman. [25]

4 Implementace

V této kapitole je uvedena vlastní implementace a výsledky použitých vybraných algoritmů: metoda neuronových sítí a metoda podpůrných vektorových strojů. Zmíněné klasifikační algoritmy jsou implementovány v open source programu RapidMiner. Jako data ke zpracování jsou použity reálné lékařské záznamy o rakovině prsu, dostupné z veřejné open source knihovny UCI repository.

4.1 Trénování a testování

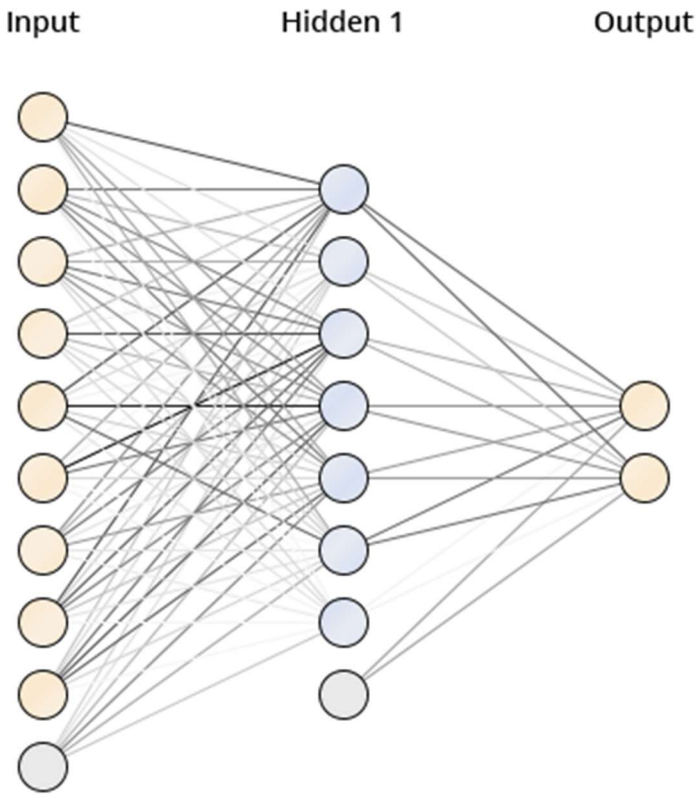
V této části je popsán postup testování algoritmů klasifikátorů. Jelikož databáze neobsahuje více souborů pro trénink a testování, ale pouze jeden soubor s uvedenými výsledky, je tento soubor rozdělen na trénovací a testovací část. A to s pomocí trojnásobné crossvalidace, jejíž aplikování provedeme v RapidMineru. Podstatou trojnásobné crossvalidace je rozdělení souboru na části. Část je použita pro testování a část pro trénování modelu. Jedná se o tzv. trénování s učitelem, neboť poskytnutá databáze obsahuje očekávané výsledky klasifikace. Očekávané výsledku budou použity pro trénink sítě a na ověření natrénovanosti algoritmu budou skryty. Celkový soubor byl rozdělen na 3 stejně velké části (subsety), tyto části byly přiděleny do jednoho ze dvou celků (tréninková a testovací data). V našem případě tedy na dva nestejně velké celky. A to na 33,3 % menší část (1 subset) a 66,3 % větší část (2 subsety). Větší část, tedy 66,6 % dat z každého subsetu bude použito pro trénování modelu, s poskytnutím informace o třídním atributu. Zbylých 33,3 % procent dat subsetu, které model ještě neviděl a jejich výsledná klasifikace třídního atributu není pro model známa, budou použity pro jeho následné testování úspěšnosti naučení. Tato operace se provede celkem tolikrát, kolik je počet subsetů, tedy 3x. Pokaždé s jinou částí daného subsetu. Abychom využili potenciál všech dostupných dat. Před použitím každého následujícího subsetu je pamět modelu vymazána. Všechna data se tedy postupně protočí jako trénovací, i jako testovací. Každá ze tří crossvalidací má za výsledek procentuální jistotu modelu v odhadu. Celková přesnost modelu je pak aritmetickým průměrem těchto tří přesností.

Toto rozdělení dat pomocí crossvalidace zamezuje přeučení modelu. To by nastalo při použití celkového množství dat pro jeho učení i pro následné testování. Přetrénování dat je nežádoucí jev, neboť model je perfektně naučen na určitý datový set, ale při předložení nových dat ke klasifikaci model ztrácí objektivitu a výsledky nejsou uspokojivé.

4.2 NN

Při implementaci byla využita funkce NN dostupná v knihovně RapidMiner. Implementovaná neuronová síť se skládá ze 3 vrstev: vstupní vrstvy, skryté vrstvy a výstupní vrstvy. Počet vstupů na vstupní vrstvě je roven počtu příznaků, které jsou rozlišovány u vstupních dat. Počet příznaků je 9. Vstupní vrstva tedy obsahuje 9 receptorů. Skrytá vrstva se skládá ze 7 neuronů. Počet neuronů skryté vrstvy byl určen RapidMinerem při optimalizaci parametrů pro dosažení nejlepšího výsledku na konkrétních datech. Počet skrytých vrstev byl manuálně předvolen na 1. Výstupní vrstva se skládá z 2 neuronů (výstupů), tento počet je roven počtu možných výstupů neboli počtu možných klasifikací vstupních dat („recidiva“ a „bez recidivy“). Architekturu sítě lze popsat 9-7-2. K natrénování sítě se využívá trénovací metoda zpětného učení- backpropagation. Jedná se o tzv. učení s učitelem.

Dále je definována aktivační funkce sigmoida ta je jako aktivační funkce volena v drtivé většině případů, proto je použita i v této práci.



Obrázek 11 Výsledná topologie neuronové sítě

K nalezení optimálních parametrů modelu sítě pro daný datový soubor byl proveden výběr a ověření parametrů neuronové sítě z možných kombinací.

V potaz byly brány tyto 4 parametry modelu:

Training cycles = počet iterací (cyklů)
Learning rate = velikost změny vah v krocích
Momentum = zajišťuje, neuváznutí funkce v lokálních minimech a gradient redukuje chybu klasifikace při jednotlivých krocích zpětného učení.
error epsilon = minimální požadované zlepšení modelu. Při nesplnění této podmínky, nebo po dokončení počtu iterací je optimalizace ukončena.

Byly provedeny výpočty pro 22506 možných kombinací v zadaném rozsahu:

Training cycles - (1; 100) v 30 krocích
Learning rate – (0,01;0,99) v 10 krocích
Momentum – (0,1;0,9) v 5 krocích
error epsilon – (0,001;1) v 5 krocích

Po každé klasifikaci vstupních dat a po každé iteraci jsou váhy upravovány na základě správnosti klasifikace. Počet kroků udává velikost změny koeficientů. Čím větší počet kroků, tím jemnější změny jsou prováděny, a tím lépe a přesněji je síť natrénována. Je tomu tak však na úkor výpočetního času.

Ověření správnosti klasifikace za konkrétního nastavení parametrů při trénování je provedeno učitelem, výsledek je porovnána s očekávaným výsledkem na výstupu a kvalitou přesnosti dosažení výsledku.

Nejúspěšnější nastavení sítě bylo vybráno na základě nejvyšší jistoty modelu ze všech provedených kombinací a následně použito jako reprezentativní v závěrečném porovnání s klasifikační metodou SVM.

Výsledek optimalizace:

Training cycles: 64

Learning rate: 0,794

Momentum: 0,34

error epsilon: 0.001

přesnost modelu: 69.47 % +/- 14.11 %

Tabulka 1 Výsledky klasifikace optimalizovaného modelu NN

	Skutečná bez-recidivy	Skutečná recidiva	Class prediction
Predikce bez-recidivy	172	58	74.78 %
Predikce recidiva	29	27	48,21 %
Class recall	85.57 %	31.76 %	

Tabulka 2 Výsledky klasifikací modelů prezentované pomocí Confusion table (vzor)

	Skutečná negativní	Skutečná pozitivní	Class prediction
Predikce negativní	TN	FN	specifita S_1
Predikce pozitivní	FP	TP	specifita S_2
Class recall	1-(FPR)	(TPR)	

Tabulka zobrazuje v kolonkách vztah mezi predikcemi a skutečnými výsledky vzhledem ke dvěma hodnotám atributů: (bez recidivy – negativní, recidiva – pozitivní)

TN – správně zvolené jako negativní (true negative)

FP – špatně zvolené jako pozitivní (false positive)

TP – správně zvolené jako pozitivní (true positive)

FN – špatně zvolené jako negativní (false negative)

Sloupec „Skutečná“ zobrazuje skutečné výsledky daného atributu

Řádek „Predikce“ zobrazuje výsledky predikované modelem

TPR = správně pozitivně hodnocené (true positive rate); (recall); (sensitivita)
Výpočet pro TPR je podíl správně pozitivně klasifikovaných k sumě pozitivních skutečností

$$TPR = \frac{TP}{TP + FN}$$

FPR = nesprávně pozitivně hodnocené (false positive rate); (false alarm); (chyba specifikace)
Výpočet pro FPR je podíl nesprávně pozitivně klasifikovaných k sumě všech negativních skutečností

$$FPR = \frac{FP}{FP + TN}$$

Přesnost modelu (accurasy) = celková průměrná přesnost modelu.
Výpočet pro accurasy je podíl správně hodnocených predikcí k celkové sumě.

$$\frac{TN + TP}{TN + TP + FN + FP}$$

Celková průměrná přesnost modelu se dá také vypočítat aritmetickým průměrem přesností jednotlivých subsetů vzniklých crossvalidací.

Ukazatel u přesnosti modelu „+/-„ (Standard deviation) je vypočítán z individuálních jistot jednotlivých subsetů. Čím menší je rozsah standard deviation, tím stabilnější je model.

Celková chyba = celková chybovost modelu.
Výpočet pro celkovou chybu je podíl nesprávně hodnocených predikcí k celkové sumě.

$$\frac{FN + FP}{TN + TP + FN + FP}$$

Specifita = procento predikcí, které byly správně klasifikovány pro konkrétní atribut

$$\mathrm{S1} = \frac{TN}{TN + FN}$$

$$\mathrm{S2} = \frac{TP}{TP + FP}$$

4.3 SVM

Při implementaci byla využita funkce SVM dostupná v knihovně RapidMiner. Byl zvolen typ SVM, který umožňuje binární lineární třídní klasifikaci. K řešení klasifikace SVM používá metodu „jeden ku jednomu“ (one-against-one). Klasifikace probíhá binárním porovnáváním mezi jednotlivými třídami.

K nalezení optimálních parametrů modelu sítě pro daný datový soubor byl proveden výběr a ověření parametrů podpůrných vektorových strojů z možných kombinací.

V potaz byly brány tyto 2 parametry modelu:

kernel type = volba jádra neboli funkce oddělující nadrovinu

parametr C (cost) = ovlivňuje přesnost klasifikace mezi třídami. Čím vyšší je hodnota parametru C, tím lépe a přesněji je algoritmus natrénovaný na konkrétní úlohu, dochází však k přeučení sítě (overfitting). Naopak čím menší je parametr C, tím méně přesnějších a výsledků algoritmus dosahuje, za méně přesného natrénování. Ale tento jev je žádoucí, neboť takový model umožňuje své aplikování na testovací data, za lepších výsledků, než přetrénovaná síť.

kernel gamma = udává jaký vliv má jednotlivý příznak na celkovou klasifikaci. Při nulové hodnotě se kernel blíží lineárnímu, naopak při vysoké hodnotě je redukován na podpůrné vektory.

Dále byly určeny podmínky, za kterých je trénování ukončeno.

max iterations = udává maximální počet provedených iterací pro trenink. Defaultně nastavený na 1000.

convergence epsilon = udává minimální požadovanou změnu chyby klasifikace mezi jednotlivými iteracemi. Defaultně nastavený na 0.001.

Trénovací algoritmus je ukončen za plnění alespoň jedné z podmínek.

Byly provedeny výpočty pro 8888 možných kombinací v zadaném rozsahu:

kernel type – 8 druhů jader
parametr C – (0,01;10) v 100 krocích
kernel gamma – (0;10) v 10 krocích

Ověření správnosti klasifikace za konkrétního nastavení parametrů při trénování je provedeno učitelem, případně je porovnána s očekávaným výstupem a kvalitou přesnosti dosažení výsledku.

Nejúspěšnější nastavení sítě bylo vybráno na základě nejvyšší přesnosti modelu ze všech provedených kombinací a následně bylo použito jako reprezentativní v závěrečném porovnání s klasifikační metodou NN.

Výsledek optimalizace:

kernel type: anova
parametr C: 0,1099
kernel gamma: 6
přesnost modelu: 71.69 % +/- 4.67 %

Tabulka 3 Výsledky klasifikace optimalizovaného modelu SVM

	Skutečná bez-recidivy	Skutečná recidiva	Class prediction
Predikce bez-recidivy	193	73	72.56 %
Predikce recidiva	8	12	60 %
Class recall	96.02 %	14.12 %	

4.4 Porovnání SVM a NN

Srovnání klasifikace pomocí neuronových sítí a podpůrných vektorových strojů jsou vizualizovány a prezentovány pomocí ROC operátoru (Reciever Operator Characteristic), který graficky porovnává klasifikace operačními křivkami. SVM a NN operační křivky jsou vypočítány z jejich Confusion matrixu (Matice záměn). Klíčové jsou hodnoty TPR a FPR. Tyto hodnoty vypočítáme pro každou jednotlivou předpověď zvlášť. Tyto průběžné výsledky jsou přeneseny jako body na osy X a Y v ROC operátoru, kde osa X reprezentuje FPR a osa Y reprezentuje TPR. Tedy z kolika procent model předpoví správně pozitivní výsledky z pozitivního celku (TPR) na ose Y. A z kolika procent model předpoví nesprávně pozitivní výsledky z negativního celku (FPR), je znázorněno na ose X. Tyto výsledky byly vypočítány v RapidMineru z každé „Matice záměn“ podle výše uvedeného vzorce.

Každá matrixová tabulka pro jednotlivé předpovědi dále udává přesnost modelu a také jistotu modelu v jeho odhadu. Průměrná přesnost modelu SVM je 71.69 %. Průměrná přesnost modelu NN je 69.47 %.

Jistota modelu pomáhá nalézt prah (threshold), zda-li chceme větší citlivost modelu, nebo naopak menší propustnost předpovědí. Záleží na zvolené taktice pozorovatele a povaze řešené problematiky.

5 Výsledky

5.1 Zhodnocení výsledků.

Z obecného hlediska jsou podstatné pro celkové zhodnocení kvality modelu nízké hodnoty FPR (chyba specifikace) na ose X a současně vysoké hodnoty TPR (senzitivita) na ose Y. Tedy co nejvíce odhadů s minimální chybovostí. V reálném použití, jako je toto, je podstatné zaměření pozorovatele na předmět zkoumání a zhodnocení nastavení prahu na základě pozorovaného subjektu.

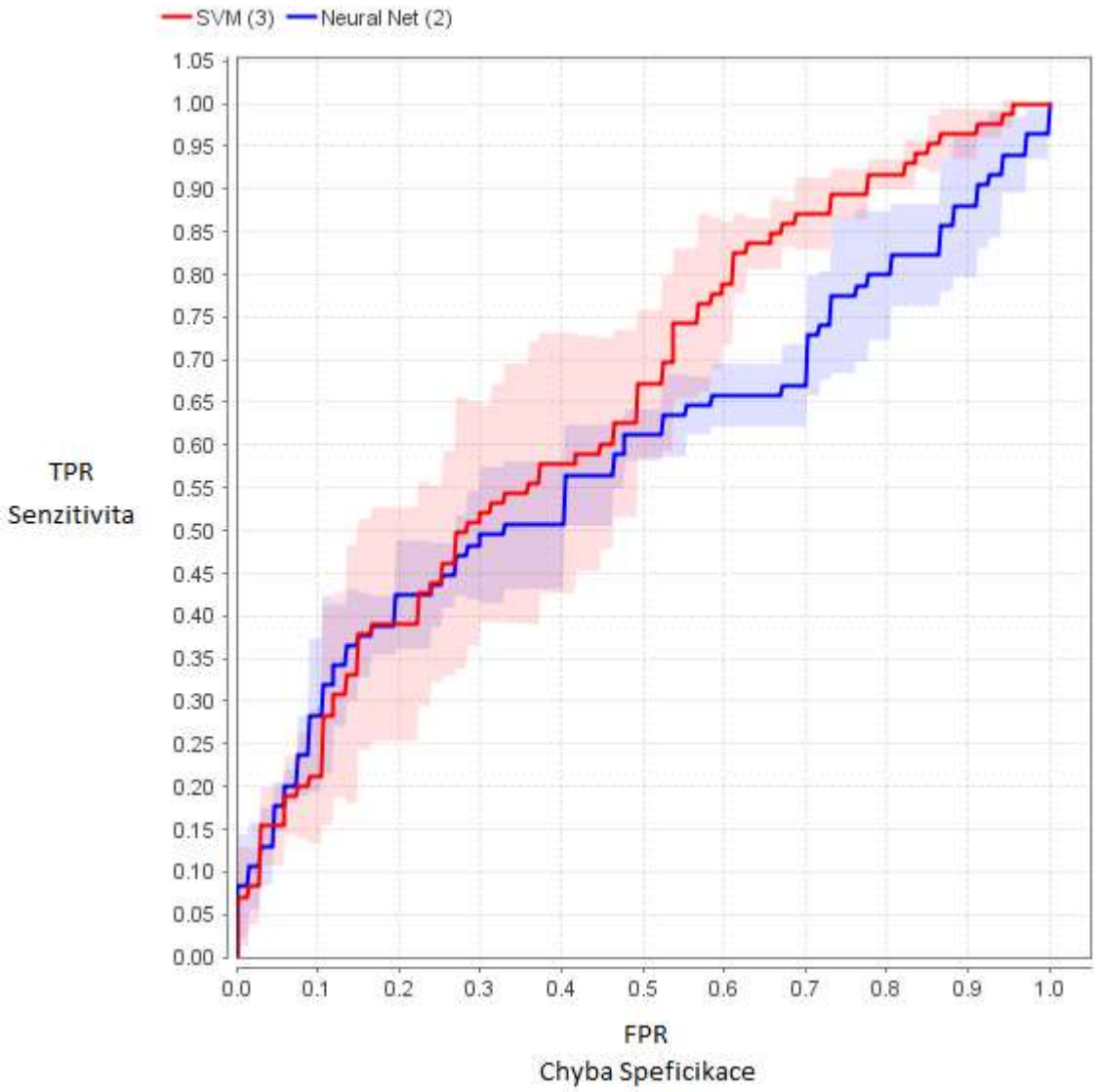
V našem případě záleží na nastavení prahových hodnot pozorovatelem. Protože hledisko obecně nejúspěšnějších výsledků není vždy nejlepší volbou. Od modelu požadujeme, aby zachytil i případy, u kterých je menší pravděpodobnost skutečného návratu choroby, ale jistá šance existuje. Proto se zaměřujeme na výsledky a porovnání modelů ve vysokých procentech FPR, tedy rozmezí 60–90 % FPR. Modely v těchto hodnotách zachycují větší množství případů, i takové, které nemusí odpovídat jistému návratu choroby. Avšak nenastane případ, u kterého by se choroba vrátila, ale model tento případ nezachytil a neoznačil jako potencionální návrat choroby. Jinými slovy případ se skutečným návratem choroby bude vždy s jistotou zachycen. Modely tedy mají prahem nastavenou vyšší citlivost. Aplikujeme tzv. „liberální přístup“.

Sledujeme proto úspěšnost algoritmů při vyšších hodnotách FPR.

5.2 Popis znázorněných výsledků.

Z porovnání ROC křivek na obr.12 plyne, že pokud požadujeme získat dostatečně vysokou senzitivita, je pro nás lepší metoda SVM. V oblasti senzitivity do 0,5 obě metody srovnatelné.

5.3 Grafické porovnání ROC



Obrázek 12 ROC křivka, porovnání SVM a NN

6 Závěr

Tato práce se zabývá klasifikačními algoritmy určenými pro identifikaci návratu nemoci podle zadaných dat o pacientovi a nemoci.

Byla provedena rešerše pro teoretická východiska a představení principů a popis fungování klasifikačních metod podpůrných vektorových strojů a neuronových sítí, vybraných pro další implementaci.

Dále byla stručně uvedena a představena využitá databáze UCI na které byly jednotlivé klasifikační algoritmy testovány. Jedná se o reálnou databázi a data jsou poskytnuta ze skutečných případů. Databáze je určena právě k tomuto následnému zpracování dat. Dále je uveden open source program RapidMiner, ve kterém probíhala klasifikace, učení a testování a díky kterému bylo vyobrazeno porovnání výsledků.

Reálná data jsme předzpracovali a importovali do RapidMineru. Tato data posloužila jako trénovací, validační a posléze i testovací soubor dat. Pomocí dvou klasifikačních algoritmů SVM a NN jsme demonstrovali klasifikaci dat. Provedli jsme optimalizaci obou klasifikačních postupů v RapidMineru na konkrétní datový soubor, abychom posléze porovnávali nejlepší možné klasifikační výsledky obou metod. Jejich srovnání jsme zobrazili pomocí ROC křivek. Na kterých byly diskutovány dosažené výsledky jednotlivých algoritmů. Klasifikátory byly testovány ve svých optimálních verzích, které byly nejúspěšnější k dané problematice. Vzhledem k nejvyšší úspěšnosti a průměrné přesnosti modelu si vedla lépe klasifikace SVM s přesností 71.69 %. Za situace posunutí prahu pozorovatelem v podmínkách blížících se reálnému prostředí, byla shledána opět klasifikace SVM jako nejvhodnější, neboť dosahovala přesnějších výsledků klasifikací za vyšší citlivosti („liberální přístup“). V obou případech dosáhla klasifikace SVM lepších výsledků.

Bakalářská práce splnila cíle v požadovaném rozsahu zadání. Námětem navazující práce by mohlo být zkoumání jiných, složitějších a obsáhlejších dat jiného typu, například pro regrese. Dále rozšíření používaných metod na podtypy implementovaných algoritmů. U SVM klasifikátoru může být provedeno aplikování dalšího přístupu „jeden proti všem“ (one-against-all) namísto „jeden proti jednomu“ (one-against-one). U neurálních sítí využít dvou skrytých vrstev místo jedné.

7 Seznam použitých zdrojů

[1] Z. Kotek, V.Chalupa, I Brůha, J.Jelínek. 1980. Adaptivní a učící se systémy. SNTL – Nakladatelství technické literatury, Praha.

[2] Z. Kotek, V. Mařík, V. Hlaváč, J. Pustka, Z. Zdráhal. 1993. Metody rozpoznávání a jejich aplikace. ACADEMIA PRAHA, Praha.

[3] O. Mikšík, 2007. Praktické využití metod digitálního zpracování obrazu: Středoškolská odborná činnost. Kroměříž: Gymnázium Kroměříž.

[4] Martin T. Hagan, Howard B. Demuth a Mark H. Baele. Orlando De Jesus. Neural network design. Boston: 1996 ISBN 05-3494332-2. Dostupné z: https://hagan.okstate.edu/NNDesign.pdf

[5] Sergios Theodoris, Konstantinos Koutroumbas. Pattern Recognition. Academic Press, 2009. ISBN 978-1-597-59749-272-0.

Dostupné z:

https://books.google.cz/books?id=gAGRCmp8Sp8C&printsec=copyright&hl=cs#v=onepage&q&f=false

[6] F. Bellakhdhar, K. Loukil a M. Abid. Face recognition approach using Gabor Wavelets, PCA and SVM [online]. International Journal of

Computer Science, 2013. Dostupné

z: http://ijcsi.org/papers/IJCSI-10-2-3-201-207.pdf

[7] Úvod do neuronových sítí. STATSOFT. [online]. 2013.

Dostupné z:

http://www.statsoft.cz/file1/PDF/newsletter/2013_02_05_StatSoft_Neuronove_site_linky.pdf

[8] BURGES, Christopher J. C. A Tutorial on Support Vector Machines for Pattern Recognition. Boston: Kluwer Academic Publishers, 1998.

Dostupné z: http://research.microsoft.com/pubs/67119/svmtutorial.pdf

[9] P. Schwarz, L. Burget, J Černocký, aj.: Klasifikace a rozpoznávání. [online], [cit. 2020-04-01]. Dostupné z: http://www.fit.vutbr.cz/study/courses/IKR/public/prednasky/

[10] Y Lecun, L.D Jackel, L Bottou, aj.: Learning Algorithms For Classification: A Comparison On Handwritten Digit Recognition. In Neural Networks: The Statistical Mechanics Perspective, editace J. H. Oh; C. Kwon; S. Cho, World Scientific, 1995, Dostupné z: http://leon.bottou.org/papers/lecun-95a

[11] P. Zemčík, R. Juránek, A Láník: Počítačové vidění. [online] Dostupné z: http://www.fit.vutbr.cz/study/courses/POV/

[12] Barlet, Marian Stewart, Javier R. Movellan a Terrence J. Sejnowski. Face Recognition by Independent Component Analysis. In: IEEE transactions on neural networks. 2002.

[13] CHANG, C.-C.; LIN, C.-J.: LIBSVM – A Library for Support Vector Machines. [cit. 2020-15-01]. Dostupné z: http://www.csie.ntu.edu.tw/~cjlin/libsvm/

[14] V. Hlaváč, 2006. Počítačové vidění a inteligentní robotika,.Praha

Dostupné z: http://people.ciirc.cvut.cz/~hlavac/teaching/MFFUK-NPGR001/Zima2018/

[15] J. Shawe, a N. Cristianini. Kernel methods for pattern analysis. Cambridge: Cambridge University Press, 2004. ISBN 05-218-1397-2.

Dostupné z:

http://read.pudn.com/downloads167/ebook/769401/Kernel%20Methods%20for%20Pattern%20Analysis.pdf

[16] Žižka, J., Internetové stránky, dostupné z: http://is.muni.cz/el/1433/podzim2006/PA034/09_SVM.pdf?fakulta=1433;obdobi=3523;kod=PA034

[17] Duda, R. O., Hart, P. E., Stork, D. G. Pattern Classification. Wiley-Interscience, New York. (2000)

[18] J. Holčík, Analýza a klasifikace dat. Akademické nakladatelství CERM, s.r.o., Brno. 2012

[19] T. Fletcher, Support Vector Machines Explained. 2009. Dustupné z: www.tristanfletcher.co.uk/SVM%20Explained.pdf

[20] WWW stránky: Support vector machine. [online], Dustupné z: http://www.statemaster.com/encyclopedia/Support-vector-machine

[21] IVANCIUC, O.: SVM - Support Vector Machines Software. [online] Dostupné z: http://www.support-vector-machines.org/SVM_soft.html

[22] M. Šonka a V. Hlaváč. Počítačové vidění. Praha: Grada, 1992.

[23] www stránky: Úvod do strojového učení (v počítačové lingvistice). [online], Dostupné z: http://wiki.matfyz.cz/wiki/Úvod_do_strojového_učení_(v_počítačové_lingvistice)

[24] Chapman & Hall, 2014. Data Mining Use Cases and Business Analytics Applications, Taylor and Francis group, LLC ISBN-13: 978-1482205497

[25] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository dostupné z: http://archive.ics.uci.edu/ml. Irvine, CA: University of California, School of Information and Computer Science. Dostupné z: https://archive.ics.uci.edu/ml/about.html

[26] Masarykova univerzita: Studijní materiály předmětu FI:PA034 [online]. 2006, Support vector machines. Dostupné z: http://is.muni.cz/el/1433/podzim2006/PA034/09_SVM.pdf?fakulta=1433;obdobi=3523;kod=PA034.

[27] MALACH, Jindřich. Rozpoznávání mluvčího na základě předepsaného textu. Brno, 1989. Kandidátská disertační práce. VUT v Brně, České Energetické Závody