ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA BIOMEDICÍNSKÉHO INŽENÝRSTVÍ

Katedra biomedicínské techniky

Automatizované zpracování EKG pomocí neuronových sítí

Automated ECG processing using artificial neural networks

Bakalářská práce

Studijní program: Klinická a biomedicínská technika
Studijní obor: Biomedicínský technik

Vedoucí práce: Mgr. Ksenia Sedova, Ph.D.

Lukáš Růžička

Kladno 2020

ČVUT
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení:	**Růžička**	Jméno:	**Lukáš**	Osobní číslo:	**474345**
Fakulta:	**Fakulta biomedicínského inženýrství**				
Garantující katedra:	**Katedra biomedicínské techniky**				
Studijní program:	**Biomedicínská a klinická technika**				
Studijní obor:	**Biomedicínský technik**				

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Automatizované zpracování EKG pomocí neuronových sítí

Název bakalářské práce anglicky:

Automated ECG processing using artificial neural networks

Pokyny pro vypracování:

Analyzujte současné možnosti softwarových nástrojů pro metody typu deep-learning, např. Microsoft Cognitive Toolkit nebo Matlab. Navrhněte a realizujte SW model s možností trénování pro využití při analýze EKG signálu. Realizujte síť pro možnost rozpoznání a predikce fibrilací. Proveďte vyhodnocení úspěšnosti detekce sledovaných situací.

Seznam doporučené literatury:

[1] Pyakillya B., Kazachenko N., Mikhailovsky N, Deep Learning for ECG Classification, Journal of Physics: Conference Series, ročník 913, číslo 012004, 2017
[2] Rozman, J., Elektronické přístroje v lékařství, ed. 1, Academia, Praha, 2006, ISBN 80-200-1308-3
[3] Metin Akay, Biomedical Signal Processing, ed. 1, Academic Press, 2012, 377 s., ISBN 0323140149
[4] Michal Janošek; Václav Kocian; Martin Kotyrba; Eva Volná, Umělá inteligence - Rozpoznávání vzorů v dynamických datech, ed. 1. vyd. , BEN-Technická literatura, 2014, ISBN 978-80-7300-4

Jméno a příjmení vedoucí(ho) bakalářské práce:

Mgr. Ksenia Sedova, Ph.D.

Jméno a příjmení konzultanta(ky) bakalářské práce:

Ing. Jan Mužík, Ph.D.

Datum zadání bakalářské práce: **17.02.2020**
Platnost zadání bakalářské práce: **19.09.2021**

prof. Ing. Peter Kneppo, DrSc., dr.h.c.
podpis vedoucí(ho) katedry

prof. MUDr. Ivan Dylevský, DrSc.
podpis děkana(ky)

PROHLÁŠENÍ

Prohlašuji, že jsem bakalářskou práci s názvem „Automatizované zpracování EKG pomocí neuronových sítí“ samostatně a použil k tomu úplný výčet citací použitých pramenů, které uvádím v seznamu přiloženém k bakalářské práci.

Nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů.

V Kladně dne 18. 5. 2020 Lukáš Růžička

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu práce paní Mgr. Ksenii Sedové, Ph.D. za vstřícnost, ochotu a trpělivost při poskytování cenných rad. Také bych rád poděkoval panu Ing. Janu Mužíkovi, Ph.D. za přínosné konzultace. V neposlední řadě bych poté rád poděkoval své rodině, která mi umožnila vystudovat.

ABSTRAKT

Automatizované zpracování EKG pomocí neuronových sítí

Cílem práce byl návrh metod strojového učení pro klasifikaci mezi fibrilujícími a nefibrilujícími pacienty na základě parametrických dat extrahovaných z EKG záznamů. Celkem byla získána data od 583 pacientů, přičemž 542 z nich fibrilaci komor neprodělalo a zbylých 42 ji prodělalo. K hodnocení byly navrženy celkem tři modely strojového učení, jeden model na principu metody logistické regrese a dvě neuronové sítě lišící se druhem aktivační funkce. Tyto modely předpokládaly rozdílné rozložení parametrických dat mezi fibrilující a nefibrilujíci skupinou pacientů. Ze získaných výsledků vyplývá, že neuronové sítě se pro účel klasifikace hodí více nežli model logistické regrese. Je navrženo využití těchto neuronových sítí jako metody, která by mohla sloužit jako dodatečná metoda predikce při použití jiných více přesných metod.

Klíčová slova

neuronové sítě, EKG, klasifikace

ABSTRACT

Automated ECG processing using artificial neural networks

The aim of the thesis was to design machine learning methods for classification between fibrillating and non-fibrillating patients based on parametric data extracted from ECG recordings, which were obtained from 583 patients, of whom 542 did not undergo ventricular fibrillation and the remaining 42 underwent ventricular fibrillation. A total of three machine learning models were proposed for evaluation, one model based on the principle of logistic regression and two neural networks differing in the type of activation function. These models assumed a different distribution of parametric data between the fibrillating and non-fibrillating groups of patients. The obtained results show that neural networks are more suitable for the purpose of classification than the logistic regression model. It is proposed to use these neural networks as a method that could serve as an additional method of prediction using other more accurate methods.

Keywords

neural networks, ECG, classification

Obsah

Seznam zkratek

Seznam zkratek

Zkratka	Význam
AL	Akutní plicní selhání (*Acute Lung Injury*)
PID	Proporcionálně-integračně-derivační
CHF	Chybová funkce
DL	Deep learning (Hluboké učení)
EKG	Elektrokardiogram
FN	False negative (Falešný negativní výsledek)
FP	False positive (Falešný pozitivní výsledek)
FPR	False positive rate (Míra falešně pozitivních výsledků)
GD	Gradient descent
HRV	Variabilita tepové frekvence (Heart rate variability)
IQR	Mezikvartilové rozmezí
KP	Klasický perceptron
LeR	Learning rate (Rychlost učení)
LiR	Lineární regrese
LR	Logistická regrese
MSE	Mean squared error
MSGD	Minibatch stochaistic gradient descent
ReLU	Rectified linear units
SGD	Stochaistic gradient desccent
SP	Sigmoidní perceptron
TN	True negative (Pravdivě negativní výsledek)
TP	True positive (Pravdivě pozitivní výsledek)
TPR	True positive rate (Míra pravdivě pozitivních výsledeků)
VF	Ventrikulární fibrilace

1 Úvod

Kardiologie je samostatný lékařský obor, který se zabývá diagnostikou a terapií srdečních a cévních onemocnění. Jedním z vyšetření, které využívá znalostí z oboru kardiologie je elektrokardiografie. Výstupem tohoto vyšetření je pak elektrokardiogram (EKG). EKG je záznam elektrické aktivity srdečního svalu v čase. Tato aktivita je projevem srdečních stahů, které jsou zajišťovány převodním systémem srdečním. EKG signál se měří elektrodami, které jsou umístěny na povrch těla v předem určených místech. Různá umístění těchto elektrod jsou označována jako svody a jejich umístění je voleno tak, aby prostorová analýza šíření vzruchu v srdci byla kompletní.

Vyšetření pomocí EKG hraje klíčovou roli v diagnostice a následné léčbě srdečních onemocnění. Pomocí analýzy křivky EKG signálu a správné interpretace zjištěných výsledků lze činit závěry o zdravotním stavu srdce pacienta. Většinu srdečních onemocnění, např. infarkt myokardu, ventrikulární tachykardie či atriální fibrilace lze diagnostikovat právě pomocí analýzy EKG signálu. V současné době je signál lidského EKG velmi dobře prostudován a jeho správná intepretace se pak stále více stává úkonem, kdy je důležité rozpoznat již dobře zažité projevy a obrazy EKG křivky. Klasifikace EKG záznamu nabývá velikého významu právě kvůli širokému rozmezí využití. Díky velkému množství vyšetření vzniká stále větší snaha proces klasifikace spolehlivě automatizovat pomocí počítačové výpočetní techniky [1].

2 Přehled současného stavu

2.1 Současný stav mortality v důsledku srdeční příhody

Zdravé srdce generuje pravidelný rytmus elektrických stahů. Pokud se tento rytmus liší od standardu, probíhá v srdci zrovna nějaký abnormální (patologický) děj, který se nazývá srdeční arytmie. Srdečních arytmií je několik typů, jedním z nich je pak ventrikulární (komorová) fibrilace (VF), která velmi často vzniká v rámci akutního infarktu myokardu. Srdeční svalové buňky nejsou nadále schopny synchronního stahu a náhodně se stahují, což má za následek tzv. „chvění komorové stěny“. V tomto stavu srdce kompletně ztrácí schopnost pumpovat okysličenou krev o oběhu a tím pádem vede ke smrti.

Celosvětově je náhlá a neočekávaná srdeční smrt nejčastější příčinou úmrtí, což představuje zhruba 17 milionů úmrtí každý rok, přičemž náhlá srdeční zástava (NSZ) představuje 25 % z nich. Přijatá definice úmrtí zapříčiněného NSZ zní takto: Smrt zapříčiněná NSZ je taková, která nastane buď do jedné hodiny od nástupu příznaků v případě, kdy tomuto nástupu příznaků byl přítomen svědek. V případě nepřítomnosti svědka je za smrt zapříčiněnou NSZ počítá smrt do 24 hodin od posledního kontaktu s živou osobou. Převážná většina úmrtí v důsledku NZS nastává právě bez přítomnosti svědka, přičemž VF je konečným stadiem srdečního mechanismu [2].

I přes to, že v posledních několika desetiletích došlo díky zlepšeným preventivním strategiím k poklesu úmrtnosti v důsledku kardiovaskulárního selhání, zvýšil se výskyt náhlých srdečních smrtí v poměru k celkovému počtu kardiovaskulárních úmrtí. K tomu došlo, jelikož celková úmrtnost v nemocnicích rychle klesla, což zdůrazňuje potřebu lepších metod klasifikace různých úrovní rizik a preventivních strategií [2]. Nejnovější data nám říkají, že 31 % všech úmrtí na světě je zapříčiněno kardiovaskulárním onemocnění a konkrétně 20 % populace pak umírá v důsledku náhlé příhody srdeční [3, 4, 5]. VF je zodpovědná za 80 % všech případů náhle příhody srdeční [6].

Maligní komorové arytmie, zejména VF, zůstávají důležitým přispěvatelem k úmrtnosti v důsledku infarktu myokardu. Úspěšnost léčby VF je stanovena časem, který uplynul mezi výskytem VF a následným poskytnutím lékařské péče. Proto hlavní strategií pro prevenci úmrtnosti v souvislosti s VF a ventrikulárními arytmiemi celkově je jejich včasná predikce [7].

2.2 Role analýzy EKG signálu za účelu predikce srdeční arytmie

Oblast medicíny se dosud silně opírala o heuristické přístupy, přičemž znalosti se v dnešní době většinou získávají prostřednictvím zkušeností a sebevzdělávání, což je ve vysoce variabilním zdravotnickém prostředí nezbytné. Nárůst znalostí a pochopení nemocí je

spojen s růstem informací a dat částečně díky pokroku v nástrojích, které generují kvantitativní, ale zároveň dostatečně kvalitativní měření fyziologických parametrů. Takto objemem rozsáhlé, a přitom dostatečně kvalitní datové pole je vhodné pro použití strojového učení (ML). Ve skutečnosti stále roste realizace potenciálu ML jako platformy, která může shromažďovat informace z mnoha zdrojů do integrovaného systému, který je poté následně schopen výrazně pomoci rozhodovacím procesům pro vysoce kvalifikované pracovníky [1].

EKG signál je velmi komplexní a často i pro odborníka těžce analyzovatelný. Ročně je odhadem nahráno asi 300 milionů EKG záznamů. S každým pacientem se daný EKG signál může podstatně lišit, a tak je jeho vyhodnocení velmi subjektivní a může dojít k chybám nebo k přehlednutí jistých informací, které se v signálu objevují. Proto je zde snaha zobjektivnit a upřesnit analýzu EKG signálu.

Jako další důvod pro vývoj automatizovaného rozhodovacího prostředku pro predikci a indikaci srdečních tachykardií je nedostatečný počet lékařů a jiných odborníků schopných správně vyhodnotit EKG záznam. V posledních letech světová populace rapidně roste a počet pacientů na jednoho lékaře se rapidně zvyšuje. Například v Indii je dnes průměrný počet pacientů připadajících na jednoho lékaře 10189, přičemž doporučovaný počet dle WHO (Světové zdravotnické organizace) je 600. Zrovna Indie je však země, ve které je ročně nejvyšší počet smrtí spojených právě s náhlou příhodou srdeční [5].

V současné době bylo již u několika parametrů extrahovaných ze signálu EKG před případným proděláním srdeční příhody prokázáno, že jejich zvýšená či snížená hodnota oproti standardu je schopna do jisté míry indikovat náchylnost k vzniku srdečních arytmií včetně VF. V několika studiích byly již zaznamenány pokusy o predikci srdečních arytmií pomocí hodnocení jednotlivých parametrů jako výskyt synkopy, systolické disfunkce levé komory, trvání QRS komplexu, disperze QT (časový interval mezi počátkem Q vlny a koncem T vlny), vyhodnocování Holterova monitorování, zprůměrovaných EKG signálů, variability srdeční frekvence, T-vln a dalších parametrů nebo metod [8, 9, 10,11,12]. Další články pak navrhují elevaci ST segmentu, přechodné rozšíření QRS komplexu a $T_{peak} - T_{end}$ interval jako nezávislé parametry schopné částečně predikovat VF [7, 13, 14]. Většina těchto studií se zaměřovala na posuzování každého takového parametru zvláště pomocí statistického zhodnocení. Posuzovány pak byly parametry před či během VF a pak parametry kontrolní extrahované z EKG záznamu zdravého jedince. Výsledky však nebyly při předpovídání fatálních událostí, jako je VT, uspokojivé. Pro vytvoření lepšího predikčního modelu VF se tedy nabízí využít více parametrů a vytvořit komplexnější klasifikátor, který dokáže řešit složité vzorce složené z těchto parametrů [3]. Zde se opět nabízí ML jako ideální platforma pro přístup k řešení takového problému.

2.3 Machine learning

Při tradičním způsobu řešení problému pomocí výpočetních technologií bylo zapotřebí naprogramovat algoritmy s přesně danými výpočetními instrukcemi a pravidly, což zpravidla vyžadovalo velmi schopného programátora s odbornou znalostí v oblasti dané problematiky. S rostoucí komplexitou řešených problémů však vyvstává problém s hledáním pravidel pro správnou evaluaci problému. S rostoucím množstvím faktorů majících, které je nutno brát v potaz, stále více a více selhává schopnost programátora stanovit taková pravidla, aby model správně fungoval. Machine Learning (ML, česky strojové učení) nabízí alternativní přístup k řešení problému. ML systém je namísto explicitního naprogramování spíše „natrénován“ [15]. ML modely získávají veškeré svoje schopnosti z dat, která jim byla předem předložena.

Počátek nového tisíciletí přinesl velký technologický progres, který vedl k výrobě velmi výkonných počítačů schopných vykonávat výpočetní úkony s extrémní rychlostí a ve velkém množství. Dalším trendem poslední doby je pak obrovský nárůst množství dat, jejichž efektivní zpracování a následná interpretace nabývá stále většího významu. ML se jeví jako jeden z vhodných kandidátů pro zpracovávání velkých množství dat, jelikož velký objem kvalitního datového pole je hlavním předpokladem pro správnou funkčnost ML modelu. [15] Moderní ML modely často vykazují lepší a přesnější výsledky než některé nejkomplexnější explicitně naprogramované programy současnosti. V některých případech byly vysoce specializované ML modely dokonce schopny předčit člověka [16].

Algoritmus strojového učení ve své nejobecnější formě lze popsat jako funkci *f(x)*, která na základě vstupního vektoru *x* vygeneruje výstupní vektor *y*. Vektor *x* o rozměru *m* lze považovat jako číselnou reprezentaci konkrétního stavu, např. hodnoty EKG parametrů jednoho pacienta, číselné hodnoty jednotlivých pixelů pro daný obrázek či hodnoty fyzických parametrů pro blíže nespecifikovaného živočicha. Vektor *y* obsahuje hodnoty, které lze považovat za predikci, kterou ML model učinil na základě vstupního vektoru *x* a implementovaného algoritmu reprezentovaného předem zmíněnou funkcí *f(x)*. Výstupní vektor *y* má potom rozměr *n*, tento rozměr není nijak závislý na rozměrech vstupního vektoru *x* [17]. V závislosti na konkrétním řešeném problému a formě výstupních dat se ML modely dělí na dva typy: klasifikační a regresní.

Klasifikační modely třídí vstupní data do dvou nebo více kategorií, výstupní vektor *y* by v tomto případě mohl například obsahovat informaci o tom, zda pacient prodělá či neprodělá fibrilaci (dvě třídy), nebo informaci o tom, jaké zvíře se nachází na obrázku. Pokud bychom těmto třídám přiřadily číselné hodnoty, lze na klasifikační model nahlížet jako na model, jehož výstupní vektor *y* obsahuje pouze diskrétní číselné hodnoty. V případě regresního modelu pak mohou složky výstupního vektoru *y* nabývat kontinuálních číselných hodnot. Příkladem může být například váha pro blíže nespecifikovaného živočicha predikovaná regresním modelem na základně vstupních hodnot fyzických parametrů.

Předtím, než ML model použijeme k predikci, je potřeba model tzv. „natrénovat“ neboli vystavit předem nasbíraným datům, která se strukturně shodují s daty, která poté bude algoritmus zpracovávat při samotné predikci. Na základě typu vstupních dat a očekávaného výstupu jsou ML trénovací algoritmy rozděleny do tří kategorií [18]:

Učení s učitelem

Při tomto typu učení je pro každý vstup (EKG parametry) již předem určen i správný výstup. Trénovací algoritmus se tak snaží nalézt pravidla a trendy v konkrétním předloženém trénovacím datovém souboru, která co nejlépe korespondují s očekávaným výstupem. Takto vytvořená pravidla poté uplatňuje při klasifikaci dat, které ještě předtím neviděl.

Učení bez učitele

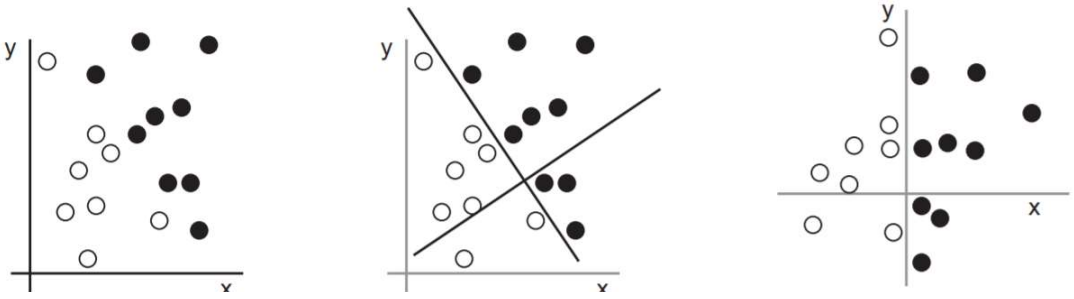
Při tomto postupu se model trénuje pouze pomocí vstupních dat. Korespondující výstupní data nejsou známa, a tak se algoritmus sám snaží data systematicky roztřídit na základě vnitřních podobností.

Zpětnovazebné učení

Tento koncept učení spadá mezi obě dříve jmenované kategorie. Model je vystaven informacím o jeho prostředí a na základě jeho akcí dostává zpětnou vazbu o tom, jaké byly výsledné hodnoty. Cílem učícího algoritmu je pak maximalizovat nebo naopak minimalizovat takového hodnoty [15].

2.4 Deep learning

Učení, v kontextu strojového učení, popisuje proces automatického vyhledávání pro lepší reprezentaci. Všechny algoritmy strojového učení spočívají v automatickém nalezení takových transformací, které mění data v užitečnější reprezentace pro daný úkol. Těmito operacemi mohou být změny systému souřadnic viz. obrázek 2.1, nebo lineární projekce, překlady, nelineární operace atd. Algoritmy strojového učení obvykle nejsou při hledání těchto transformací kreativní a pouze prohledávají předdefinovanou sadu operací, které se nazývají prostor hypotéz [15].



Obrázek 2.1: Transformace systému souřadnic, převzato z [15]

ML je tedy technicky vyhledávání užitečných reprezentací některých vstupních dat v rámci předdefinovaného prostoru možností pomocí zpětnovazebního signálu. Deep learning (DL, česky hluboké učení) je specifickým dílčím oborem ML. Jeho podstatou je nový pohled na učení reprezentací z dat, který klade důraz na učení následných vrstev stále významnějších reprezentací. Za slovem deep (česky hluboké) se v kontextu DL skrývá myšlenka několika po sobě jdoucích vrstev reprezentací. Čím více těchto vrstev je přítomno v modelu, tím je tento model „hlubší“. Zatímco u klasického ML model pracuje většinou s jednou až dvěma vrstvami reprezentací, moderní DL často zahrnuje desítky nebo dokonce stovky po sobě jdoucích vrstev reprezentací, které jsou všechny automaticky stanoveny po interakci s tréninkovými daty. V DL jsou tyto vrstvené reprezentace z drtivé většiny učeny prostřednictvím modelů nazývaných neuronové sítě.

2.4.1 Neuronové sítě

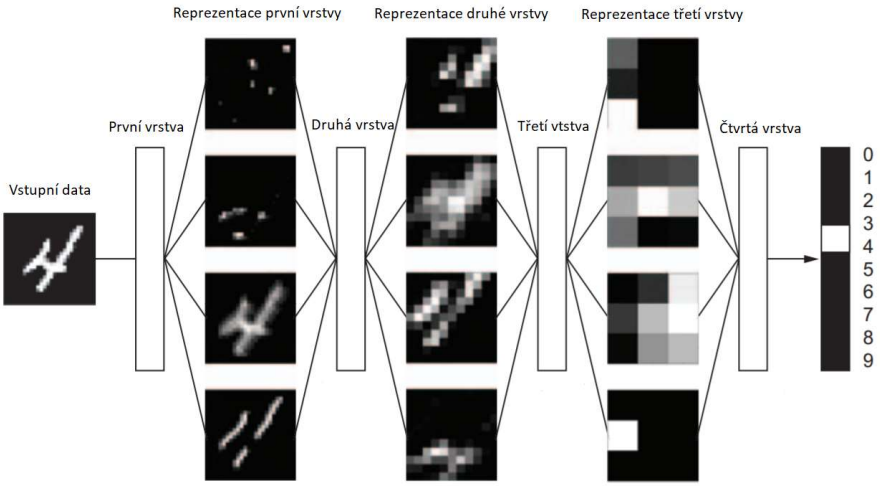
Existuje několik typ neuronových sítí, přičemž volba typu při vytváření DL modelu závisí na charakteru řešeného problému. Základní principy a uspořádání neuronových sítí však zůstává pro všechny typy identická. Obsahem této práce bude implementace tzv. „artificial neural network“. Struktura této sítě a princip jejího fungování budou detailně popsány v metodice. V této podkapitole si pouze stručně popíšeme, jakým způsoben se snaží neuronová sít reprezentovat data.

Velkou výhodou neuronové sítě je její schopnost samostatně najít opakující se vzory či trendy, které systematicky dekódují vstupní data. Tyto trendy často nemusejí dávat na první pohled velký smysl a nelze je tak odvodit z teoretických znalostí dané problematiky. Proto se neuronové sítě často používají ke systematické kategorizaci dat, které nelze jednoduše roztřídit podle předem daných pravidel, která určí teorie.

I odborníci si často nemusí být jisty, podle kterých pravidel se nakonec neuronová síť po natrénování chová. Obrázek 2.2 však celkem dobře ilustruje, jakým způsobem lze mechanismus neuronové sítě chápat z velmi širokého úhlu pohledu. Vysvětluje sice princip na příkladu klasifikace obrázku ručně psaného čísla, avšak tento způsob uvažování lze aplikovat při klasifikaci jakéhokoliv typu dat, a ne pouze ke klasifikaci čísel.

Jak je vidět na obrázku 2.2, síť transformuje číselný obraz na reprezentace, které se stále více liší od původního obrazu a stále více informují o konečném výsledku. Hlubokou síť lze považovat za vícestupňovou operaci destilace informací, kde informace procházejí po sobě jdoucími filtry a na výstupu vychází stále více a více „vyčištěny“ [15]. Jak již bylo předem zmíněno, programátor takovéto neuronové sítě má však malý vliv na to, jakým způsobem si daný algoritmus bude vytvářet tyto reprezentace, a tak i v případě obrázku 2.2 se jedná pouze o teoretickou úvahu a výsledný algoritmus může pak nakonec pracovat s úplně odlišnými reprezentacemi. Je za potřebí míti na vědomí, že neuronová síť se snaží najít v datech jakákoliv pravidla, pomocí kterých je schopna třídit tyto data

do jednotlivých skupin či odhadovat kontinuální hodnoty výstupu. Na rozdíl od teoretika se neřídí pravidly problematiky, a tak i konkrétní reprezentace nemusí leckdy s teoretickými hodnotami vůbec souviset.



Obrázek 2.2: Způsob reprezentace neuronovou sítí, převzato z [15], upraveno

2.5 Machine learning pro vyhodnocování EKG signálu

Pro detekci arytmie bylo navrženo velké množství klasifikátorů. Navrhované techniky sahají od jednoduchých klasifikátorů, jako jsou lineární klasifikátory nebo rozhodovací stromy, až po sofistikovanější, jako jsou tradiční neuronové sítě, Support Vector Machines nebo podmíněné náhodné pole. Kromě toho bylo mnoho prací věnováno nalezení nejlepší kombinace funkcí, někdy dokonce vývoji komplexních metod zpracování signálu, a výběru nejlepší podskupiny (zmenšení rozměrů) pro klasifikaci arytmie. Na jedné straně jsou populární volbou vstupních funkcí morfologické znaky extrahované z časové oblasti (jako jsou intervaly mezi dvěma vlnami (např. interval R-R), amplitudy, obsah oblasti pod křivkou), funkce ve frekvenčních doménách, vlnková transformace, komplexní reprezentace srdečního rytmu nebo statistika vyššího řádu. Na druhé straně metody výběru prvků, jako je analýza nezávislých složek, analýza hlavních složek, optimalizace rojů částic nebo genetický algoritmus neuronové sítě, tzv. back – propagation [1, 19].

Vstupními daty v mém případě byly parametry extrahované z časové domény, frekvenční domény a nelineární analýzy. Bylo již použito několik typů neuronových sítí s rozdílnou volbou takových parametrů, u kterých je z fyziologického hlediska prokázáno, že mají přímou či nepřímou spojitost s danou srdeční arytmií. Jejich přesnosti, sensitivity a specificity se pohybují kolem 90-95 % [3, 5, 6, 20]. Všechny tyto neuronové

sítě pracovali s časovými informace jako HRV (heart rate variability, česky variabilita srdeční frekvence) nebo délka R-R intervalu. Žádné z nich nepoužili prediktory, které v sobě neobsahují časovou informaci, např. amplituda T vlny. Takovéto parametry (zmíněné již výše) byly však statisticky prokázány za schopné predikovat VF.

3 Cíle práce

Cílem této práce je navrhnout postup pro automatické zpracování a následné vyhodnocení EKG signálu, kdy je snahou klasifikovat jednotlivé záznamy EKG signálu pro různé pacienty a úspěšně predikovat, zda u daného pacienta proběhne či neproběhne fibrilace komor. Existuje již několik parametrů, u kterých byla prokázána statistické významnost při predikci VF. Dává smysl tedy předpokládat, že tyto parametry by se mohly prokázat jako vhodná vstupní data pro ML modely. V klinické praxi lze tyto parametry extrahovat z EKG signálu.

Nejprve je zapotřebí ověřit, zda vstupní data jsou vhodná pro klasifikaci a zda číselné parametry, které byly předem extrahovány z časového průběhu signálu mají výpovědní hodnotu. K tomu poslouží statistické testy pro odlišné distribuce skupin dat. Extrahovaná parametry musí z fyziologického hlediska souviset s možností vzniku ventrikulární fibrilace a musí být statisticky rozlišitelné.

Dále je cílem navrhnout několik ML modelů pro predikci VF a následně pak zhodnotit jejich přesnost. Při výběru vývojového softwaru a metody je potřeba brát v potaz komplexnost EKG signálu a strukturu vstupních dat.

4 Metody

Aby bylo možno splnit cíle této práce, bylo nejprve zapotřebí ověřit vhodnost dat EKG záznamů pomocí vhodné statistické analýzy. Vstupní data bylo nejprve potřeba statisticky zhodnotit a zjistit tak, zda jsou vhodná pro implementaci ML metod. Aby byl ML algoritmus schopen spolehlivé klasifikace dat mezi dvěma skupinami (fibrilující a nefibrilující pacienti), je potřeba, aby hodnoty jednotlivých parametrů tyto dvě skupiny měly odlišné statistické rozdělení či tvořili shluky. Proto byla data analyticky testována na rozdílné rozložení.

Na základě výsledků statistických testů bylo vytvořeno několik setů dat pro účely trénování ML modelů. Následně byly navrženy celkem tři ML klasifikační metody, jejichž výkonnost byla následně porovnána. Nejdříve byla aplikována ML metoda logistické regrese a následně pak dvě neuronové sítě lišící se strukturou.

4.1 Předzpracování dat

Původní dataset obsahoval data 583 pacientů se 20 parametrickými hodnotami, jejichž přehled je uveden v tabulce 4.1. Tyto data byla získána od nejmenované švédské nemocnice. Jednalo se o číselné parametry extrahované z EKG signálů jednotlivých pacientů, které byly nahrávány v průběhu jedné hodiny před potencionálním proděláním VF. Jednotky pro parametrické hodnoty nebyly v původním datasetu uvedeny, a tudíž v průběhu práce nejsou uvedeny.

Data bylo potřeba nejdříve předzpracovat z důvodu chybějících záznamů či neadekvátních hodnot, které by do výsledného vyhodnocení ML modely mohly zanést chybu. Neadekvátními hodnotami jsou myšleny takové hodnoty, které v souboru nahrazovali chybějící záznam. Rozdílné označení chybějících záznamů bylo zapříčiněno z důvodu sběru dat z více odlišných přístrojů, které pracují s jinými datovými formáty či softwary. Byli vyřazeni ti pacienty, u kterých data pro všechny parametry nebyla kompletní či jeden a více parametrů nabývaly neadekvátních hodnot. Po vyřazení těchto chybných záznamů nakonec výsledný dataset sestával z dat pro 408 pacientů. Shrnutí procesu předzpracování je uvedeno v tabulce 5.1.

4.2 Testování odlišného rozdělení parametrů

Před implementací ML algoritmů bylo potřeba nejdříve ověřit, zda se data liší svým rozložením. Rozdílné rozložení je důležitý předpoklad pro úspěšnou klasifikaci pomocí ML či DL modelů. Jednotlivé parametry byly testovány zvlášť a v případě, kdy neprokázaly dostatečně rozdílné rozložení, byly z datasetu vyřazeny. Pro následnou statistickou analýzu byl použit statistický software SPSS Statistics 23.0 vyvinutý společností IBM.

Nejprve byla data podrobena Kolmogorov-Smirnově testu normality. Konkrétně byla použita tzv. „one-sample“ verze tohoto testu, jejímž principem je srovnávání rozložení testovaného parametru s předem daným typem rozložení, v tomto případě s normálním typem rozložení. Ani u jednoho parametru nebylo prokázáno normální rozložení. Z tohoto důvodu byla data prezentována pomocí hodnot mediánů a mezikvartilových rozmezí (IQR).

Následně byl použit neparametrický Mann-Whiney U test. Hladina významnosti α byla zvolena jako 5 %. Nulová hypotéza v tomto případě tvrdila, že distribuce daného testovaného parametru se statisticky neliší pro skupinu fibrilujících a nefibrilujících pacientů, hypotéza alternativní toto tvrzení zamítala. Výsledky Mann-Whitneyho U testu pro jednotlivé parametry jsou uvedeny v tabulce 5.2. Pro grafické znázornění rozdílných distribucí jsou grafy těchto distribucí uvedeny v příloze B.

Tabulka 4.1: Přehled extrahovaných parametrů

Popis parametru	Zkratka
Maximální amplituda T vlny	Tamp_max
Minimální amplituda T vlny	Tamp_min
Rozdíl maxima a minima T vlny	Tamp_disp
Interval mezi nejdříve detekovaným počátkem a nejpozději detekovaným koncem T vlny přes všech 12 svodů	G_TpkTend
Obsah pod křivkou ST segmentu	summaST
Maximální amplituda ST segmentu	max_ST
Průměrná hodnota ST segmentu	OverallQRS
Srdeční frekvence	HR
Obsah po křivkou T vlny pro Wilsonovy svody	Tarea_V1-V6
Obsah po křivkou T vlny pro Einthovenovy svody	Tarea_I-III
Obsah po křivkou T vlny pro všechny Goldbergovy svody	Tarea_aVL, aVR, aVF

4.2.1 ROC analýza

Následně byla provedena ROC analýza pro jednotlivé parametry. Výstupem analýzy byly ROC křivky (ROC křivka je popsána níže), pomocí jichž byly určeny hodnoty tzv. „cutoff“ hodnoty thresholdy (viz. níže). Tyto hodnoty byly následně použity pro transformaci kontinuálního typu parametrických dat na binární. Hodnotám vyšším, než

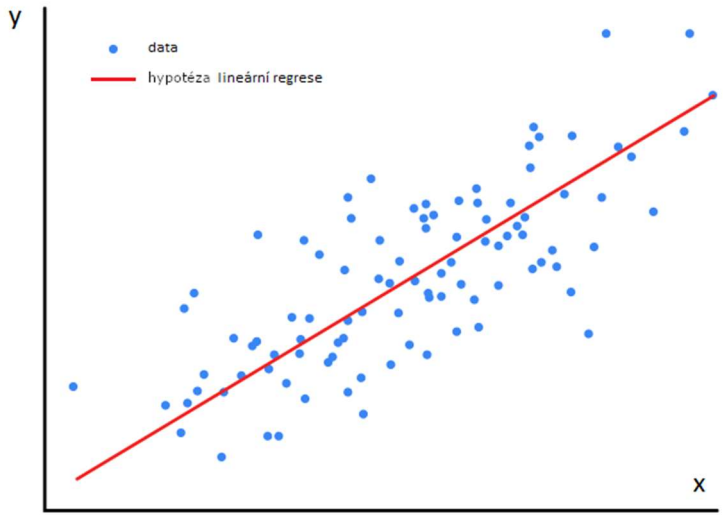
hodnota cutoff byla přiřazena hodnota 1 a nižším hodnota 0. Grafy jednotlivých ROC křivek jsou uvedeny v příloze A.

4.3 Logistická regrese

První metodou, která byla implementována, byla logistická regrese (LR). Jedná se o ML metodu učení s učitelem, která pracuje s kontinuálním formátem vstupních dat a následně je pak třídí do dvou a více kategorií. V případě, kdy výstupní data sestávají pouze ze dvou kategorií, nazývá se ML model binominální (z anglického slova binominal) LR či pouze LR. V případě více než dvou kategorií se používá anglický název multinominální (z anglického slova multinominal) LR. Pro klasifikaci EKG dat byla použita binominální LR.

4.3.1 Základní princip lineární regrese

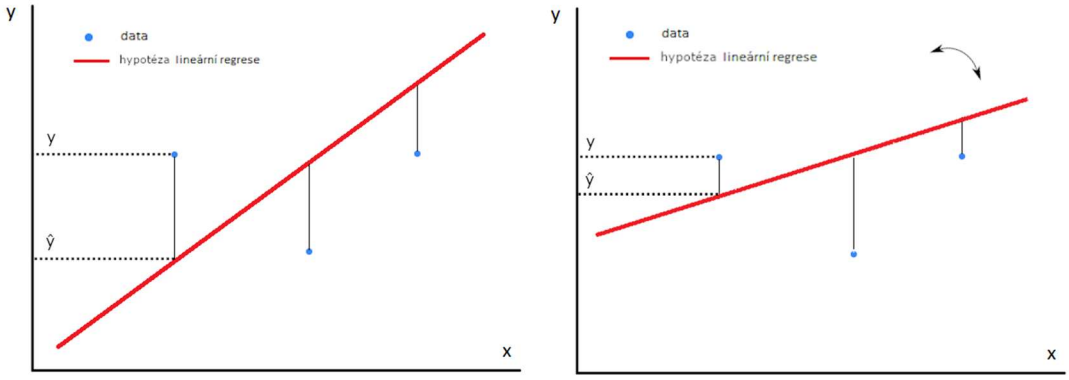
K LR by se dalo přistupovat jako k modifikované verzi tzv. „lineární regrese“ (LiR). LiR je nejjednodušší a nejrozšířenější ML technika pro predikci výstupních hodnot na základě kontinuálních vstupních dat. Při tvoření LiR modelu se snažíme najít způsob, jak pomocí přímky vysvětlit vztah mezi závislou proměnnou (hodnota, kterou se model snaží předpovědět) a jednou nebo více vysvětlujícími proměnnými (vstupní data sloužící jako prediktory). V případě pouze jednoho prediktoru lze model LiR znázornit na následujícím obrázku 4.1. V této práci se pracuje s více prediktory, pro názornost je však problematika LR vysvětlena na příkladu datasetu s jedním prediktorem, jelikož lze tím pádem využít 2D grafu pro příkladné znázornění. Pro více parametrů p lze uplatnit stejnou logiku, přičemž grafy by nabývaly (n-1) počtu dimenzí.



Obrázek 4.1: Model lineární regrese, převzato z [21].

Funkce či křivka, pomocí jíž se ML model snaží modelovat danou situaci se nazývá hypotéza. Z obrázku 4.1 (červeně) je tedy vidět, že hypotéza LiR modelu má tvar přímky, která je zvolena tak, aby co nejvíce vystihovala trénovací data. Proces, při kterém se zvolí taková orientace přímky, aby co nejvíce vystihovala trénovací data se nazývá „fittování“. Cílem fittování je minimalizace tzv. „chybové funkce“ (CHF). Hodnota této funkce poskytuje informaci o tam, jak přesně model vystihuje data, na kterých byl natrénován. Podrobně je CHF vysvětlena v kapitole 4.5.1.

V tuto chvíli je důležité si uvědomit, že cílem fittování je minimalizovat hodnotu CHF. Jelikož čím více se budou hodnoty $\hat{y}$ predikované modelem lišit od skutečných hodnot y, tím menší bude hodnota CHF viz rovnice (4.13). Na obrázku 4.2 je znázorněno, jakým způsobem probíhá fittování hypotézy LiR pomocí algoritmického výpočtu MSE. Počáteční orientace křivky je zvolena náhodně.



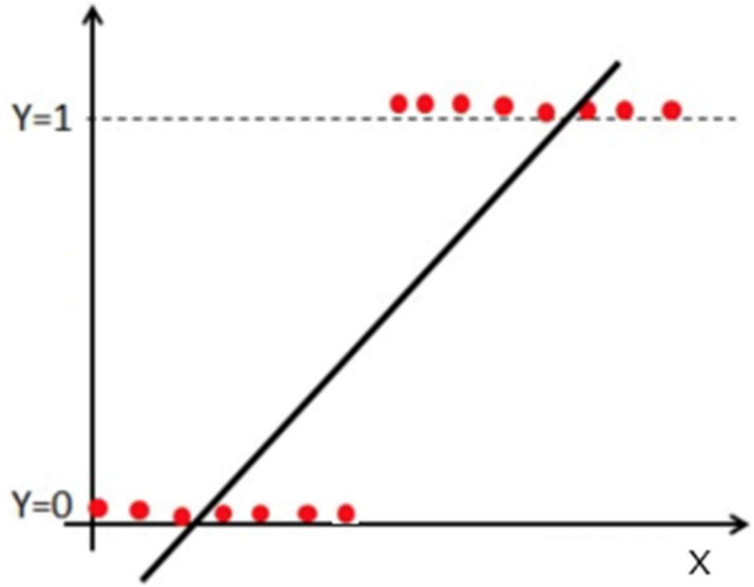
Obrázek 4.2: Proces fittování, autor

4.3.2 Implementace logistické regrese

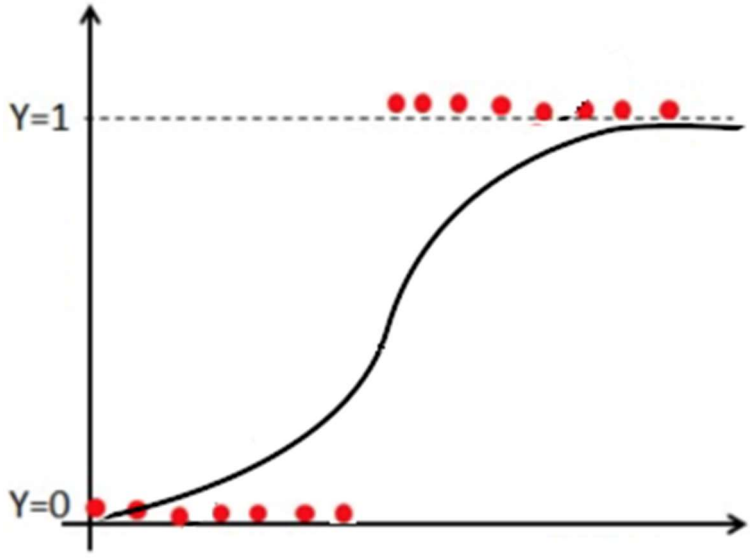
Slovo regrese v názvu LR naznačuje, že se nejedná o klasifikační algoritmus v pravém slova smyslu. Model totiž určuje pravděpodobnost, se kterou konkrétní instance patří do třídy označené číslem 1 (fibrilující pacienti). Vertikální osa grafu tedy poté nabývá hodnot v intervalu <0;1>. Dále se pak hodnotám výstupního vektoru *y* přiřadí hodnota 0 pro negativní výsledek (v případě EKG nefibrilující pacienti) a hodnota 1 pro pozitivní výsledek (fibrilující pacienti). Toto přiřazování je určeno pouze konvencí.

V případě implementace LiR na binominální klasifikační problém vyvstávají hned dva problémy, které jsou zřetelně vidět na obrázku 4.3. Zaprvé hypotéza není schopná dostatečně vymezit hranici mezi dvěma třídami (nefibrilující a fibrilující pacienti). Zadruhé, jelikož tvarem hypotézy LiR je přímka, její obor hodnot nabývá hodnot v intervalu <-∞; ∞>, což je nežádoucí, jelikož pro predikci pravděpodobnosti nemá smysl uvažovat hodnoty mimo interval <0;1>. Tyto problémy jsou vyřešeny změnou tvaru hypotézy použitím funkce „sigmoid“. Tato funkce a veškeré její parametry jsou pospány v kapitole 4.4.2 – neuronové sítě. Funkce sigmoid je vhodná právě proto, že jednak se její

obor hodnot pohybuje v intervalu <0;1>, zároveň její tvar je schopen lépe vystihnout přechod mezi dvěma třídami. Hypotéza logistické regrese je zobrazena na obrázku 4.4.



Obrázek 4.3: Lineární regrese implementovaná na binominální klasifikační problém, převzato z [22], upraveno



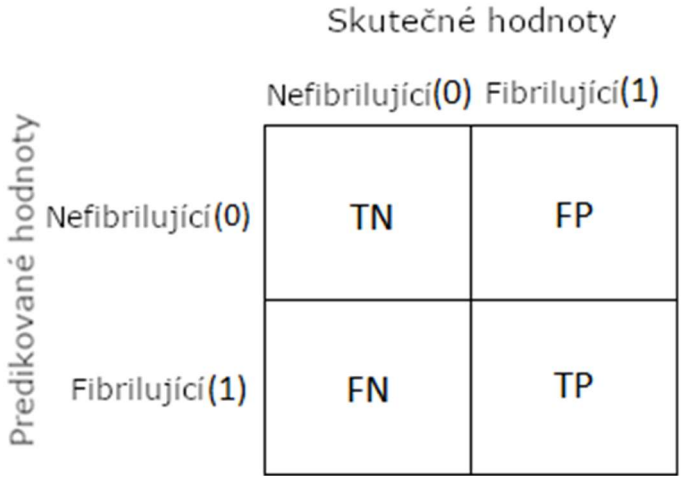
Obrázek 4.4: Hypotéza logistické regrese, převzato z [22], upraveno

4.3.3 Určení mezní hodnoty (threshold)

Pokud tedy LR model předpovídá pravděpodobnost, se kterou daná instance patří do třídy $y = 1$ (fibrilující pacienti), je potřeba zvolit mezní hodnotu této pravděpodobnosti. Tato mezní hodnota se nazývá „threshold“ a jako výchozí hodnota se, pokud není explicitně

specifikováno, používá 0,5. Znamená to tedy, že pokud bude pravděpodobnost, že daný pacient patří do třídy fibrilujících, větší než 50 %, pak ho model do této třídy zařadí a naopak. V některých případech je však vhodnější zvolit jinou hodnotu thresholdu.

Při vytváření ML modelu pro predikci VF je více důležité, aby byl model schopen správně predikovat co nejvíce VF i na úkor toho, že by měl u několika pacientů předpovědět VF, která nakonec neproběhne. V případě správné pozitivní predikce lze použít termín „true positive“ (TP), v případě nesprávné pozitivní predikce pak „false positive“ (FP). Dalšími dvěma termíny pak jsou „true negative“ (TN) (správná predikce, že pacientovi nehrozí VF) a „false negative“ (FN) (model nesprávně predikuje, že pacientovi nehrozí VF). Přehled těchto pojmů je na obrázku 4.5, který zobrazuje tzv. „confusion“ matici, která se používá jako jedna z metod vyhodnocení přesnosti modelu. V jednotlivých polích se poté nacházejí počty klasifikovaných instancí. V případě více než dvou tříd by matice nabývala rozměrů $n \times n$, kdy n je počet tříd. Pro stoprocentně přesný model se budou veškeré hodnoty nacházet na uhlopříčce z levého horního rohu do pravého dolního rohu. Veškeré hodnoty ležící mimo tuto osu poté snižují přesnost modelu.



Obrázek 4.5: Confusion matice, autor

Dále lze poté přesnost ML modelu určit pomocí několika metrických parametrů, kde každý z nich vypovídá o výkonnosti ML modelu z trochu jiného úhlu pohledu. Krom přesnosti se také jedná o tzv. „specificitu“, „senzitivitu“ a „preciznost“. Všechny tyto parametry nabývají hodnot v intervalu <0;1> a obvykle se vyjadřují v procentech. V kontextu medicíny podává specificita informaci o tom, s jakou přesností je model schopen určit pacienty s pozitivním výsledkem na test (pacienti s VF – TP). Senzitivita je číselné vyjádření schopnosti modelu správně identifikovat pacienty s negativním výsledkem (pacienti bez VF – TN). Preciznost podává informaci o tom, jak často ML model chybuje při klasifikaci pacientů patřících do fibrilující skupiny, resp. jak často

zařadí nefibrilujícího pacienta do fibrilující skupiny. Rovnice pro výpočet přesnosti, senzitivity, specificity a preciznosti jsou následující:

$$\text{Přesnost} = \frac{\text{TN} + \text{TP}}{\text{TN} + \text{TP} + \text{FN} + \text{FP}} \tag{4.1}$$

$$\textit{S}\text{enzitivita} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.2}$$

$$\textit{S}\text{pecificita} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{4.3}$$

$$\text{Preciznost} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.4}$$

4.3.4 ROC křivka

Pro určení hodnoty thresholdu se používá tzv. ROC křivka, která popisuje vztah mezi specificitou a senzitivitou, resp. mezi specificitou a tzv. „false positive rate“ (FPR), kterou lze spočítat dle rovnice:

$$\text{FPR} = 1 - \text{specificita} \tag{4.5}$$

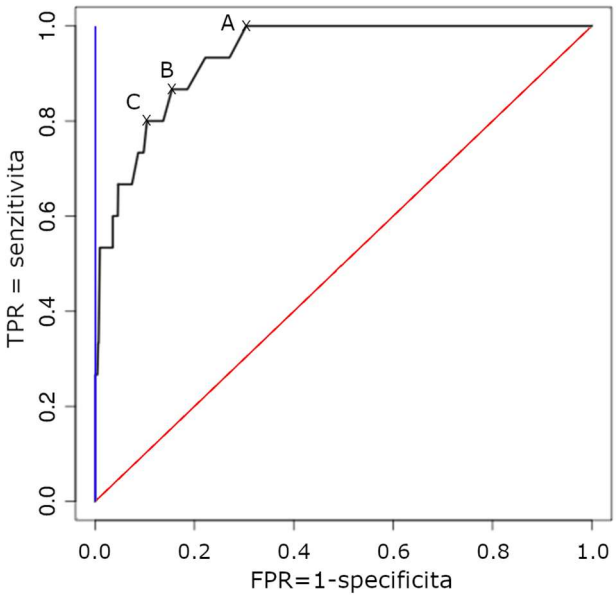
a po dosazení z rovnice (4.3):

$$FPR = 1 - \frac{TN}{TN + FP} \tag{4.6}$$

FPR tedy vyjadřuje, jak hodně je model náchylný ke špatné klasifikaci pacienta s VF. ROC křivka je poté definována jako závislost mezi FPR na horizontální ose a „*true positive rate*“ (TPR), která je shodná se senzitivitou, na ose vertikální. Příklad ROC křivky je vidět na obrázku 4.6.

Jednotlivé hodnoty ROC křivky jsou vypočítány na základě různých hodnot thresholdu (rozmezí v intervalu <0;1>). Pro dokonalý model by ROC křivka měla tvar vertikální úsečky shodné s vertikální osou (na obrázku modře). Naopak ROC křivka modelu bez schopnosti předpovídat výsledky, který by kategorizoval nefibrilujícími a fibrilujícími pacienty zcela náhodně (tedy pro obě kategorie s 50% šancí), by měla tvar diagonální úsečky (na obrázku červeně). Na základě tvaru ROC křivky lze poté dle potřeby volit hodnotu thresholdu podle toho, jakou od modelu požadujeme senzitivitu na úkor případných falešně označených fibrilujících pacientů. V případě ROC křivky na obrázku 4.6 bychom volili hodnotu thresholdu pro bod A, kdybychom chtěli zajistit

maximální specificitu ML modelu na úkor zvýšeného počtu FPR. Většinou je však vhodné hledat kompromis mezi TPR a FPR, zde by se jako vhodné hodnoty thresholdů jevily např. hodnoty pro body B či C.



Obrázek 4.6: ROC křivka, převzato z [23], upraveno

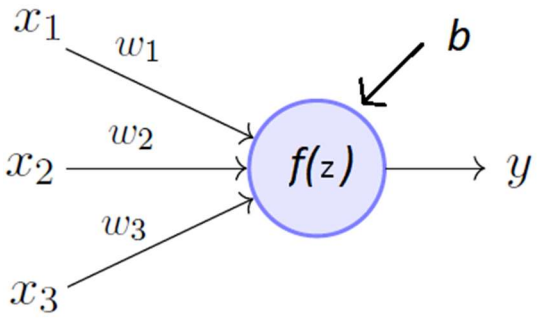
4.4 Neuronové sítě

Jednou z mnoha výhodných vlastností umělých neuronových sítí je možnost analýzy velkého množství dat za účelem určení vzorů a charakteristických znaků, a to zejména v případech, pro které nejsou k dispozici matematické vztahy, případně jsou dostupné matematické vztahy velmi vzdálené od požadované přesnosti, či v případech, kde s ohledem na charakter zpracovávaných informací a dat nelze aplikovat metody konvenční statistiky. Umělé neuronové sítě mají, mimo jiné, dvě zcela zásadní schopnosti, tj. schopnost rozpoznávání a učení, což umožňuje řešení složitých úloh z problematiky predikce a rozhodování. Na rozdíl od statistických analýz s více proměnnými, jež se dříve pro tuto problematiku používaly, mají umělé neuronové sítě naprosto jedinečnou vlastnost, resp. umí odhalit vztahy a vazby, zatímco regresní či diskriminační analýzy ze své podstaty vyžadují povahu skrytých vztahů znát. Další výhodou umělých neuronových sítí je fakt, že se dokáží adaptovat na neúplná data. Právě tato vlastnost umožňuje jejich použití pro velmi složité aplikace jako je například rozpoznávání obrazu. Jinými slovy lze tvrdit, že příspěvek každého jednotlivého zpracovávaného elementu není zásadně důležitý, což v důsledku znamená, že chování umělé neuronové sítě ani výsledek nejsou významně ovlivněny, přestože v datech chybí některé elementy. V neposlední řadě, umělé neuronové sítě umí odhadnout jak kvantitativní proměnné, tak proměnné třídy (někdy

také označované jako statické proměnné), proto je lze využít pro účely predikce a klasifikace [24].

4.4.1 Perceptron

Základními funkčními jednotkami neuronové sítě jsou tzv. „perceptrony“. Jedná se o jednotky, které mají v kontextu neuronových sítí představovat jednotlivé neurony. Klasické perceptrony (KP) přijímají vstupní hodnoty *x* a pomocí jistých matematických operací je převádějí na výstupní hodnoty *y*. Vstupní hodnoty *x* i výstupní hodnoty *y* mohou nabývat pouze binárních hodnot (0 nebo 1). Vstupy mohou pocházet buď ze vstupní vrstvy (vstupní data), nebo z KP v předchozí vrstvě. Schéma KP je vidět na obrázku 4.7:



Obrázek 4.7: Schéma perceptronu, převzato z [25], upraveno

Proces výpočtu hodnoty vstupu *y* nejprve začíná sečtením vážených vstupních hodnot (součin každé hodnoty vstupu z předchozí vrstvy *x* vynásobené jejím váhovým koeficientem *w*). Od této sumy se poté následně odečte hodnota *b* zvaná „bias“. Pro takovýto vážený součet *z* poté platí:

$$z = \sum_{i=1}^{N} (x_i \cdot w_i) - b \tag{4.7}$$

kde N je počet vstupních hodnot *x*. Váhové koeficienty *w* společně s koeficientem *b* hrají velmi důležitou roli v trénovacím procesu. Při trénování se ML model snaží identifikovat, jak důležitou roli hrají jednotlivé vstupní hodnoty *x* na konečný výsledek. Pomocí postupného upravování jednotlivých hodnot w_i je model schopen zvýšit či naopak potlačit vliv jednotlivých vstupních hodnot x_i na konečnou hodnotu výstupu *y*. Hodnotu biasu *b* lze interpretovat jako „prahovou“ hodnotu konkrétního perceptronu, přičemž čím více se zvyšuje hodnota *b*, tím více se také zvyšuje citlivost perceptronu. Tedy platí:

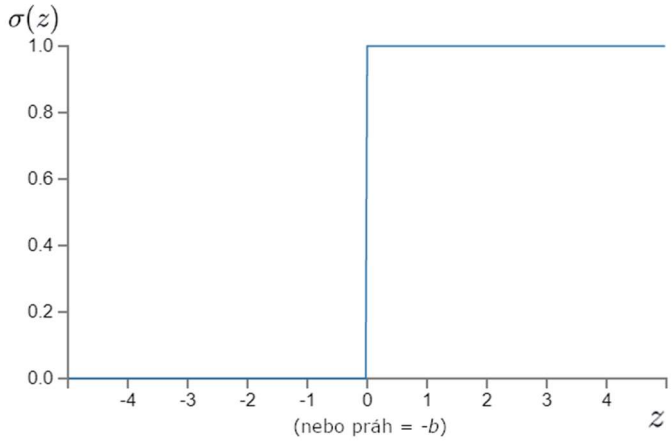
$$y = \begin{cases} 0 \text{ pokud } \sum_i x_i \cdot w_i + b \leq 0 \\ 1 \text{ pokud } \sum_i x_i \cdot w_i + b \geq 0 \end{cases} \tag{4.8}$$

respektive:

$$y = \begin{cases} 0 \text{ pokud } \sum_i x_i \cdot w_i \leq práh \\ 1 \text{ pokud } \sum_i x_i \cdot w_i \geq práh \end{cases} \tag{4.9}$$

kde *práh* představuje vlastně zápornou hodnotu biasu, tudíž pro něj platí, že čím vyšší je jeho hodnota, tím nižší je citlivost perceptronu a následně jeho ochota k „aktivaci“ (hodnota $y = 1$).

Na schématu KP (obrázek 4.7) je ještě znázorněna tzv. „aktivační funkce“ $f_{(x)}$. Existuje mnoho druhů aktivačních funkcí, které se používají z důvodu standardizace a vnesení většinou jiné, než lineární závislosti mezi váženým součtem z a výstupem y. V případě KP je použita jednoduchá skoková aktivační funkce, přičemž umístění pozice „skoku“ je dáno prahovou hodnotou, viz obrázek 4.8.

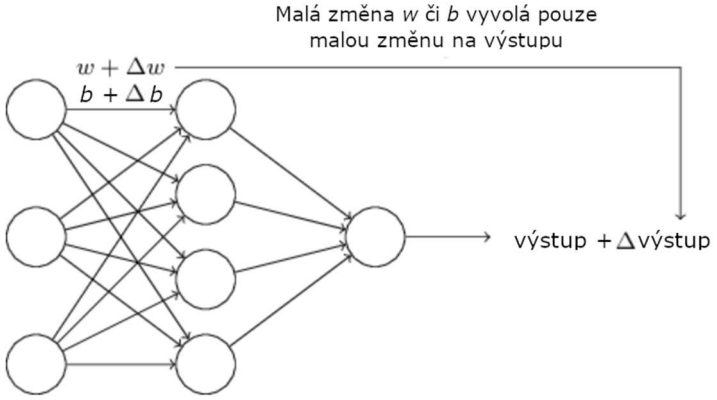


Obrázek 4.8: Skoková aktivační funkce, převzato z [26], upraveno

4.4.2 Sigmoidní perceptron

ML model složený z KP nelze však jednoduše natrénovat a optimalizovat. Předpokládejme, že máme ML model složený z výše uvedených KP, který bychom chtěli použít, abychom se naučili řešit nějaký konkrétní problém (v našem případě budou vstupy do ML modelu parametrická data z EKG záznamu). Při procesu trénování takového modelu bychom chtěli, aby daný model dokázal určit takové hodnoty pro jednotlivé

váhové koeficienty *w* a hodnoty biasu *b*, které by zajistily přesnou a konzistentní klasifikaci mezi fibrilujícími a nefibrilujícími pacienty. Chtěli bychom tedy docílit toho, aby po provedení malé změny (jemné změny hodnoty *w* či *b)* byla vyvolána pouze malá změna i na výstupu. Tato vlastnost je klíčová pro proces trénování ML modelu a ilustruje jí obrázek 4.9.



Obrázek 4.9: Vztah mezi změnou hodnot *w,b* a změnou hodnoty výstupu, převzato z [26], upraveno

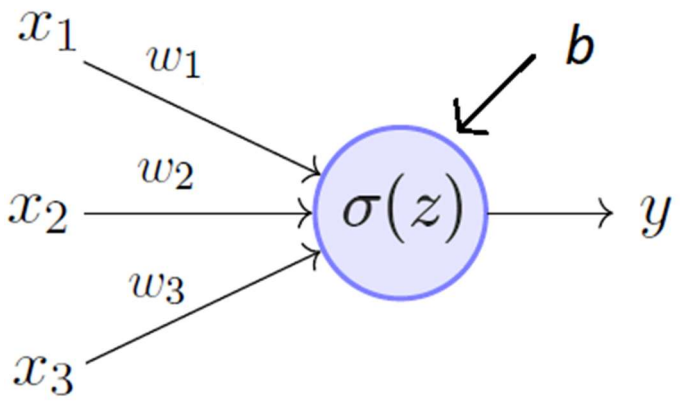
Za předpokladu, že malá změna váhového koeficientu *w* či biasu *b* způsobí pouze malou změnu na výsledném výstupu ML modelu, mohli bychom tuto skutečnost použít k úpravám *w* a *b* s úmyslem, aby se naše síť chovala více způsobem, jak chceme. Tato skutečnost by pak poskytovala ML modelu možnost lépe reagovat na tyto změny.

Předpokládejme například, že ML model omylem klasifikoval fibrilujícího pacienta jako nefibrilujícího. Kdybychom přišli na to, jakým způsobem provést malou změnu koeficientů *w* a *b* tak, aby byl ML model o trochu více náchylnější ke klasifikaci dotyčného pacienta jako fibrilujícího, mohli bychom potom proces klasifikace znovu zopakovat, vyhodnotit a zase upravit hodnoty koeficientů. Postupným opakování výše zmíněného sledu procesů bychom se stále více blížili k situaci, kdy by ML model nakonec vyhodnotil dotyčného pacienta jako fibrilujícího, obecně řečeno: hodnota výstupu by byla stále přesnější a více by modelovala konkrétní problém. ML model by se tímto způsobem učil.

Problém je v tom, že výše zmíněný proces učení nejsme schopni realizovat, pokud by měl být ML model složen z KP. Kdyby tomu tak bylo, mohla by malá změna váhového koeficientu *w* nebo biasu *b* jakéhokoli jednotlivého perceptronu v síti způsobit, že se výstupní hodnota *y* tohoto perceptronu zcela převrátí, od 0 do 1 či naopak. Jelikož jsou v ML modelu perceptrony mezi sebou spojeny a mají vzájemně se ovlivňují, toto náhlé skokové převrácení by mohlo nadále způsobit radikální změnu v chování zbytku perceptronů v rámci ML modelu. Takto by se ML model sice přiblížil ke správné klasifikaci jednoho konkrétního pacienta, je postup při klasifikaci ostatních pacientů

v trénovacím datasetu by se však nějakým obtížně kontrolovatelným způsobem úplně změnil. Z výše uvedeného tedy vyplývá, že učení ML modelu složeného z KP je prakticky nemožné.

Tento problém můžeme překonat zavedením nového typu perceptronu, tzv. sigmoidního perceptronu (SP). Tento typ perceptronu je v mnoha ohledech podobný KP, ale je modifikován tak, aby malé změny jeho váhových a bias koeficientů způsobily pouze malou změnu hodnoty jeho výstupu. Toto je klíčová vlastnost, která umožní trénování ML modelů. SP lze schématicky zobrazit podobně jako KP (obrázek 4.10):



Obrázek 4.10: Schéma sigmoidního perceptronu, převzato z [25], upraveno

Mezi KP a SP existují dva důležité rozdíly. Prvním rozdílem je, že hodnota výstupu *y* pro SP nemusí nabývat pouze binárních hodnot, nýbrž se pohybuje v intervalu <0;1>. Z logiky věci poté vyplývá, že jelikož SP mohou být v rámci různých ML modelů za sebou řetězeny, tak i pro vstupní hodnoty *x* platí, že mohou také nabývat hodnot v intervalu <0;1>, protože výstupní hodnota perceptronu v předchozí vrstvě je vstupní hodnota pro perceptron v následující vrstvě.

Druhým rozdílem mezi KP a SP je odlišná aktivační funkce. Jak už z názvu vyplývá, aktivační funkce SP se nazývá „sigmoid“ σ. Sigmoid funkce je zobrazena na obrázku 4.11 a je definována jako:

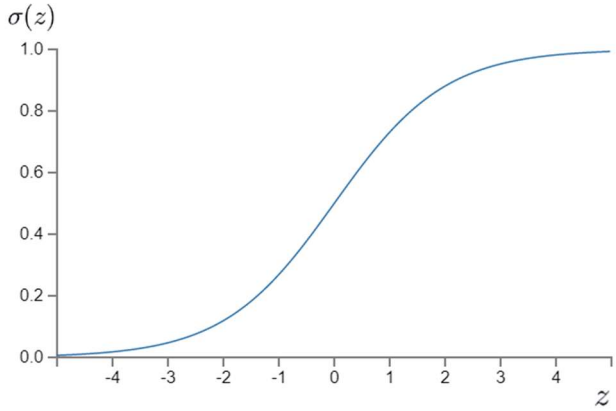
$$\sigma_{(z)} = \frac{1}{1 + e^{(-z)}} \tag{4.10}$$

a po explicitním vyjádření váženého součtu *z*:

$$\sigma_{(z)} = \frac{1}{1 + e^{(-\sum_i x_i \cdot w_i - b)}} \tag{4.11}$$

I přes výše zmíněné odlišnosti se SP chová v některých případech skoro stejně jako KP. Pro porozumění předpokládejme, že hodnota váženého součtu *z* bude nabývat velmi

vysoké hodnoty. V tomto případě platí, že $e^{(-z)} \approx 0$ a tím pádem $\sigma_{(z)} \approx 1$. Pomocí stejné logiky naopak platí, že pro velmi nízké hodnoty z bude $e^{(-z)} \to \infty$ a tím pádem $\sigma_{(z)} \approx 0$. Je Tedy v obou případech extrémních hodnot váženého součtu z se SP a KP chovají téměř identicky. Pouze pokud vážený součet z dosahuje průměrných hodnot, tak se model SP odchyluje od modelu KP.



Obrázek 4.11: Sigmoid funkce, převzato z [26], upraveno

Z obrázku 4.11 je vidět, že funkce sigmoid je vlastně „vyhlazená“ forma skokové funkce (viz. obrázek 4.8). Právě tento hraje zásadní roli ve funkci SP a je důvodem, proč je SP schopen na malé změny váhových koeficientů *w* a biasu *b* reagovat malou změnou výstupní hodnoty *y*. Dále pak platí, že přibližnou hodnotu změny výstupní hodnoty Δy lze spočítat jako:

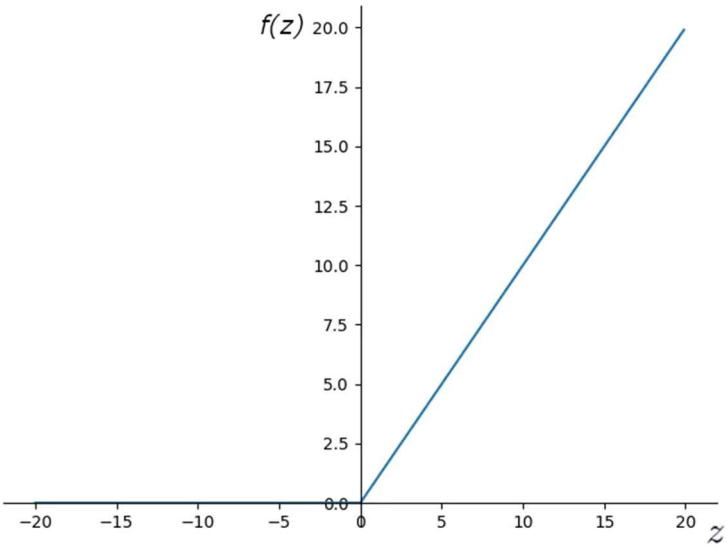
$$\Delta y \approx \sum_i \frac{\partial y}{\partial w_i} \Delta w_i \frac{\partial y}{\partial b_i} \Delta b_i \tag{4.12}$$

Z výše uvedeného vzorce je patrné, že změna výstupní hodnoty Δy je lineární funkcí váhových koeficientů *w* a biasů *b*. Tato lineární závislost umožňuje postupné přizpůsobování *w* a *b* koeficientů, jelikož ekvidistantní změny těchto koeficientů zapříčiní ekvidistantní změny výstupní hodnoty SP.

Díky možnosti hodnoty výstupu *y* nabývat libovolných hodnot v intervalu <0;1> je možné implementovat SP do ML modelu pro řešení regresního problému, kdy predikce nabývají kontinuálních číselných hodnot. V případě binominálního klasifikačního problému se určí prahová hodnota výstupu, tzv. „threshold“ (většinou 0,5), která představuje hranici mezi dvěma třídami. Pokud bude výstupní hodnota ML modelu pro predikci fibrilace nabývat například hodnoty 0,37, pak ML model klasifikuje daného pacienta jako nefibrilujícího (za předpokladu, že třída nefibrilujících pacienti je označena číslem 0 a třída fibrilujících pacientů číslem 1. Problematice určování optimální hodnoty thresholdu se věnuje text v kapitole 4.3.4.

4.4.3 ReLU

ReLU (z angl. „rectified linear units“) je další varianta aktivační funkce, která může být implementována do ML modelu. Jedná se o variantu aktivační funkce, která má silné biologické a matematické opory. V roce 2011 bylo prokázáno, že její použití zlepšuje přesnost tréninkového procesu neuronových sítí. Funguje na principu pouze jedné prahové hodnoty $f_{(z)} = 0$ pro všechny hodnoty váženého součtu z menší než 0. Jednoduše řečeno, výstupem ReLU je nula ($f_{(z)} = 0$) když $z < 0$, a naopak pro hodnoty $z > 0$ je na výstupu lineární funkce [27] . Funkce ReLU je znázorněna na obrázku:

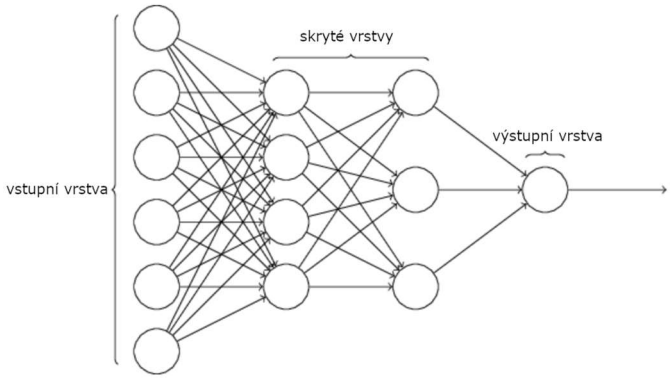


Obrázek 4.12: ReLU funkce, převzato z [28], upraveno

Obecným problémem u sigmoidní funkce je to, že se saturuje. To znamená, že všem velkým hodnotám váženého součtu z je přiřazena hodnota výstupu 1 a všem malým hodnota 0. Z tohoto důvodu je funkce pouze citlivá na změny kolem svého středu vstupu, například 0,5. K omezené citlivosti a saturaci funkce dochází bez ohledu na to, zda sumarizovaný součet z na vstupu obsahuje užitečné informace nebo ne. Jakmile je výstupní hodnota nasycena, stává se pro ML algoritmus náročné přizpůsobovat váhové a bias koeficienty pro zlepšení výkonu modelu.

4.4.4 Struktura neuronové sítě

Na obrázku 2.2 byla obrazně zobrazena struktura neuronové sítě. Z obrázku lze vyčíst, že neuronová síť se skládá z vrstev (z angl. „layer“), které jsou postupně skládány za sebou. Tyto vrstvy jsou složeny z perceptronů, přičemž dělíme vrstvy na vstupní, skryté a výstupní. Základní schéma neuronové sítě je vidět na obrázku 4.13.



Obrázek 4.13: Základní schéma neuronové sítě, převzato z [26], upraveno

Vstupní vrstva je obsahuje perceptrony, které do sítě vnáší informace dané příslušných datasetem. I když jsou tyto perceptrony značeny ve schématu shodně s ostatními, je důležité si uvědomit jejich odlišnost. Vhodnější je smýšlet o těchto perceptronech jako u „buňkách“, jejichž výstupní hodnotou *y* jsou číselné hodnoty jednotlivých instancí z datasetu. Neprobíhá zde žádný z procesů jako u ostatních perceptronů (výpočet váženého součtu, transformace přes aktivační funkci atd.).

Výstupní vrstva obsahuje perceptrony, na základě jejichž výstupu se ML model rozhoduje. V případě binominální klasifikace fibrilujících pacientů se jedná pouze o jeden perceptron, jehož výstup lze interpretovat jako ano – dotyčný pacient prodělá VT (výstupní hodnota překročila prahovou hodnotu) či ne – fibrilace neproběhne (výstupní hodnota nepřekročila prahovou hodnotu).

Střední vrstvy se nazývají skryté vrstvy, protože perceptrony v této vrstvě nejsou ani vstupy ani výstupy. Perceptrony v těchto vrstvách zpracovávají a transformují vstupní data, která jsou poté předána výstupní vrstvě. Zatímco počet perceptronů ve vstupní a výstupní vrstvě neuronové sítě je zcela zřejmý (je dán počtem parametrů extrahovaných z EKG signálu pro vstupní vrstvu a jeden perceptron pro ano/ne klasifikaci ve výstupní vrstvě), množství skrytých vrstev a počet perceptronů v těchto jednotlivých vrstvách je spíše záležitostí heuristického přístupu, jelikož proces návrhu skrytých vrstev nelze jednoduše shrnout pomocí několika základních pravidel.

V kontextu s obrázkem 4.13 je důležité mít na paměti, že každý perceptron má pouze jednu výstupní hodnotu, která je následně v sítí posílána dále ke všem perceptronům následující vrstvy. Z estetických důvodů se však ve schématech neuronových sítí zobrazuje tato skutečnost jako několik samostatných výstupů, z nichž každý vede k jednomu perceptronu v následující vrstvě.

4.5 Proces optimalizace

Optimalizace je považována za nejdůležitější složku v procesu navrhování ML algoritmů. Výběr optimalizačního algoritmu může znamenat rozdíl mezi dosažením dobré přesnosti v hodinách nebo dnech. Většina ML algoritmů zahrnuje optimalizaci nějakého druhu. Optimalizace v kontextu ML odpovídá úkolu minimalizace nebo maximalizace některé funkce $f_{(x)}$ pomocí postupné změny hodnoty x. U většiny optimalizačních problémů se jedná o minimalizaci hodnoty $f_{(x)}$. Případné maximalizace lze dosáhnout pomocí algoritmu minimalizace opačné hodnoty ($-f_{(x)}$). V této práci se je cílem minimalizace této funkce [29].

4.5.1 Chybová funkce

Prvním krokem při optimalizaci ML modelu je definice výše zmíněné funkce $f_{(x)}$. Tato funkce se nazývá tzv. „chybová funkce“ (CHF) či „ztrátová funkce“ a v anglické literatuře, která je ve srovnání s českou nesmírně rozsáhlejší, se označuje hned několika názvy – „cost function“, „loss function“ či „error function“. Dle konvence se většinou značí písmenem *J*.

ML modely se učí pomocí CHF. Jedná se o metodu hodnocení, která poskytuje informaci o tom, jak dobře specifický algoritmus modeluje dané údaje. Pokud by se předpovědi učiněné ML modelem příliš lišily od skutečných výsledků, CHF by vykazovala velmi velké číslo. Čím více by se předpovězené hodnoty blížily hodnotám skutečným, tím více by se snižovala hodnota CHF.

Standardní LF, která se používá jako výchozí pro většinu ML modelů, je tzv. „mean squared error“ funkce (MSE). Rovnice pro výpočet MSE je uvedena níže:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y} - y)^2 \tag{4.13}$$

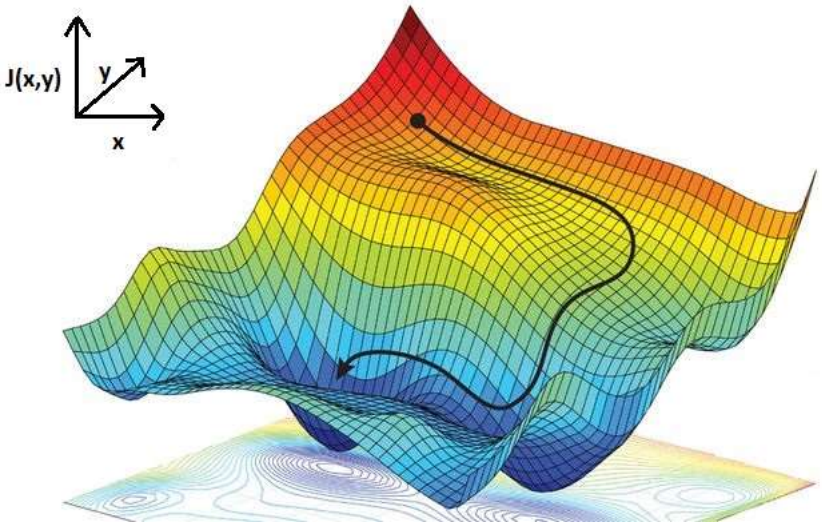
kde N je počet trénovacích hodnot, y je výstupní hodnota trénovacího datasetu a $\hat{y}$ je predikce této hodnoty učiněná ML metodou.

Druhá mocnina výsledné chyby (rozdíl mezi predikovanou hodnotou $\hat{y}$ a skutečnou hodnotou y ve vzorci (4.13) je použita hned ze dvou důvodů. Zaprvé zaručuje, že rozdíl bude nabývat pouze kladných hodnot. Kdyby v konkrétním bodě x_i ležela hodnota y (na obrázku 4.2 modrý bod) nad hodnotou hypotézy $\hat{y}$ (na obrázku 4.2 červená křivka), chyba by poté nabývala záporné hodnoty. V opačném případě by chyba nabývala naopak kladné hodnoty. Druhým umocněním chyby se tady eliminuje možnost vzájemného odečtení chyb s opačným znaménkem v procesu sumace. Druhá mocnina pak také zaručuje, že

křivka MSE bude mít konvexní tvar, čehož se využívá při optimalizaci ML modelu např. pomocí tzv. „gradient descent“ metody, která je podrobně popsána níže.

4.5.2 Gradient descent

Cílem optimalizačního algoritmu je najít hodnoty parametrů hypotézy (viz. níže) daného ML, které odpovídají minimální hodnotě CHF. Základním principem optimalizační metody „gradient descent“ (GD) je výpočet prvních parciálních derivací pro jednotlivé parametry ML hypotézy vzhledem k CHF. Hodnoty parciálních derivací následně poskytují informaci, jakým směrem učinit „krok“. V případě trojrozměrného grafu CHF si lze metodu GD představit jako simulaci malého tělesa, které se v důsledku gravitačních sil bude pohybovat směrem dolů viz. obrázek 4.14.



Obrázek 4.14: Princip metody gradient descent, převzato z [30], upraveno

V rámci GD metody lze rozlišovat mezi třemi různými variantami, lišícími se frekvencí iterace. Klasická přístup má stejnojmenný název GD a při něm dochází k aktualizace (ke kroku) CHF až po výpočtu parciálních derivací všech instancí v tréninkovém datasetu. Proces optimalizace pomocí klasické metody GD může být velmi zdlouhavý, pokud trénovací dataset obsahuje velké množství instancí.

Dalšími dvěma variantami jsou poté metody „stochaistic gradient descent“ (SGD) a „minibatch stochaistic gradient descent“ (MSGD). SGD aktualizuje hodnotu CHF okamžitě po výpočtu parciálních derivací pro jednu konkrétní instanci a metoda MSGD aktualizuje hodnotu po výpočtu pro několik instancí, tato skupina instancí se označuje jako „batch“ (várka), přičemž počet instancí ve batchi lze měnit, standardní velikostí je 32 instancí. Výhodou SGD a MSGD je jejich mnohem větší rychlost oproti klasické metodě GD na úkor menší přesnosti minimalizace CHF. Schopnost minimalizace pomocí

SGD a MSGD však většinou pro účely ML modelu postačuje, a proto bude použita MSGD i v této práci.

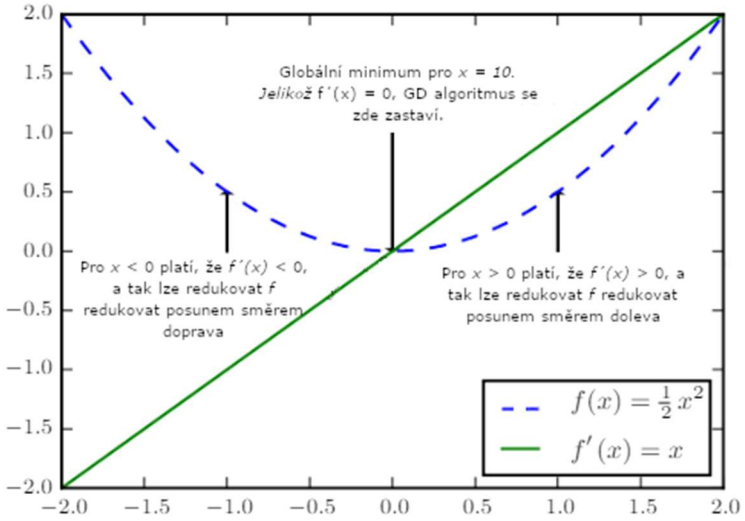
4.5.3 Matematický základ metody gradient descent

Předpokládejme, že máme funkci $y = f_{(x)}$, kde x a y jsou reálná čísla. Derivace této funkce označíme jako $f'_{(x)}$. Derivace $f'_{(x)}$ udává sklon $f_{(x)}$ v bodě x. Jinými slovy specifikuje, jakým způsobem učinit malou změnu na vstupu abychom získali odpovídající změnu ve výstupu dle této rovnice:

$$f_{(x+\varepsilon)} \approx f_{(x)} + \varepsilon \cdot f'_{(x)} \tag{4.14}$$

kde ε je hodnota iteračního kroku.

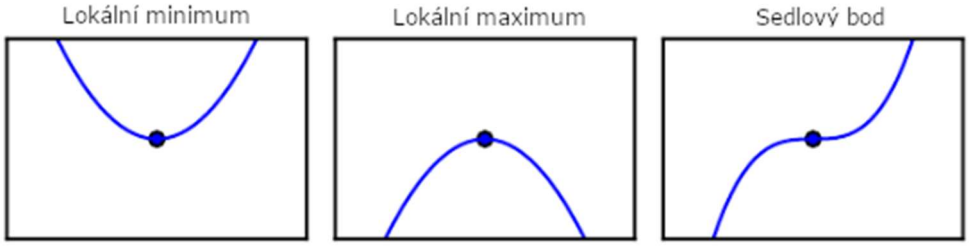
Z tohoto důvodu je derivace užitečná pro minimalizaci funkce, jelikož nám říká, jakým způsobem změnit x, aby došlo k malému zmenšení CHF a tím pádem přiblížení predikovaných hodnot ke skutečným. Můžeme tedy tvrdit, že například výraz $f(x - \varepsilon \cdot sign(f'_{(x)}))$ je menší než $f_{(x)}$ pro dostatečně malé ε, což nám dovoluje snížit hodnotu $f_{(x)}$ pohybem x v malých krocích s opačným znaménkem derivace. Názorná ukázka takového postupu je zobrazena na obrázku 4.15 [29].



Obrázek 4.15: Proces nalezení globálního minima pomocí výpočtu gradientu, převzato z [29], upraveno

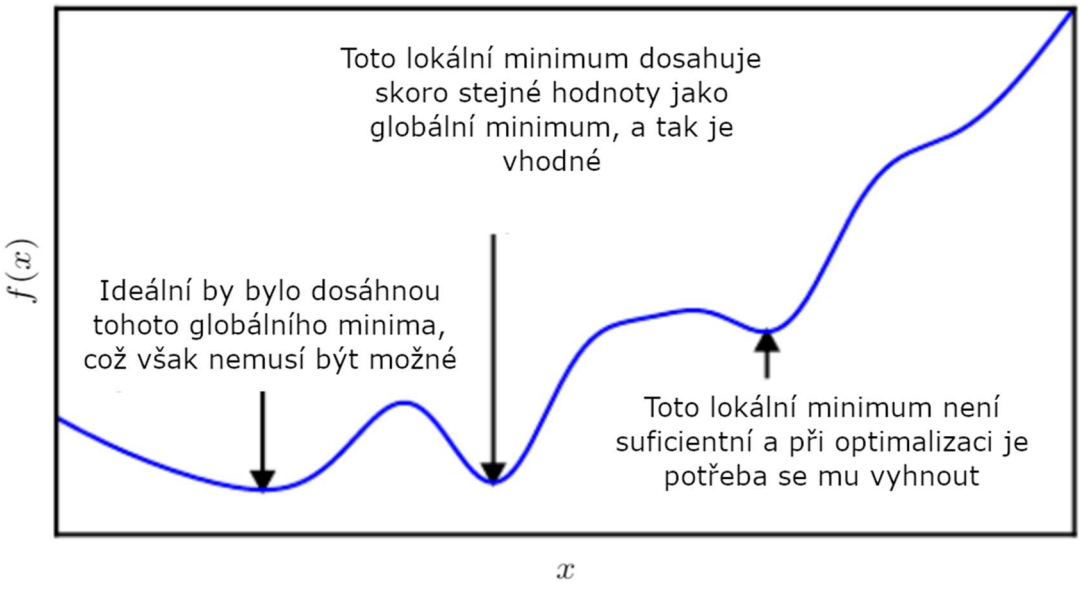
Pokud $f'_{(x)} = 0$, derivace neposkytuje žádnou informaci o tom, kterým směrem se pohybovat. Body, pro které platí, že $f'_{(x)} = 0$, se nazývají jako kritické body nebo stacionární body. Lokální minimum je bod, kde hodnota $f_{(x)}$ je nižší než ve všech sousedních bodech, takže již není možné dále její hodnotu snižovat pomocí pohybu po malých krocích. Stejně tak lokální maximum je bod, kde hodnota $f_{(x)}$ je vyšší než ve všech

sousedních bodech, tudíž není možné zvýšit její hodnotu pohybem po malých krocích. Některé kritické body nejsou ani maxima, ani minima, a přesto je hodnota derivace rovna nule. Tyto body poté označujeme jako sedlové body. Příklady každého typu kritického bodu lze vidět na obrázku 4.16.



Obrázek 4.16: Kritické body, převzato z [29], upraveno

Bod, ve kterém hodnota $f_{(x)}$ nabývá absolutně nejnižší hodnoty z celého definičního oboru $f_{(x)}$, se nazývá globální minimum. Jedná se o ideální bod, do kterého cílíme při procesu optimalizace. Může existovat pouze jedno globální minimum, zatímco lokálních minim může být hned několik. Je také možné, že existují lokální minima, která nejsou globálně optimální. V rámci ML optimalizujeme funkce, které mohou mít mnoho lokálních minim, které nejsou optimální, a mnoho sedlových bodů obklopených velmi rovnými regiony. To vše ztěžuje optimalizaci, zvláště když je funkce $f_{(x)}$ vícerozměrná. Proto se obvykle snažíme nalézt takovou hodnoty funkce $f_{(x)}$, která je velmi nízká, ale ne nutně globáně minimální [29]. Viz obrázek 4.17 pro znázornění.



Obrázek 4.17: Jednotlivé kritické body v rámci jedné funkce, převzato z [29], upraveno

4.5.4 Hyperparametry

V kontextu ML modelu lze rozlišovat mezi parametry a tzv. „hyperparametry“. Parametry jsou proměnné, které jsou interní pro daný model a jejichž hodnotu lze odhadnout či zvolit na základně datasetu, se kterým ML model pracuje. Jinými slovy se jedná o proměnné, které jsou ML modelem určeny v procesu tréninku a modelují vztah mezi vstupními a výstupními daty. Příkladem mohou být například hodnoty váhových koeficientů či biasu.

Hyperparametry ML modelu jsou místo toho vlastnosti, které řídí celý tréninkový proces. Zahrnují proměnné, které určují strukturu sítě (například počet skrytých vrstev) a proměnné, které určují, jak je síť trénována (například rychlost učení). Hyperparametry se nastavují před procesem tréninku ML modelu a jejich hodnoty jsou určovány samotným programátorem na základě heuristických pravidel či na základě experimentování a srovnávání dosažených výsledků pro různé hodnoty.

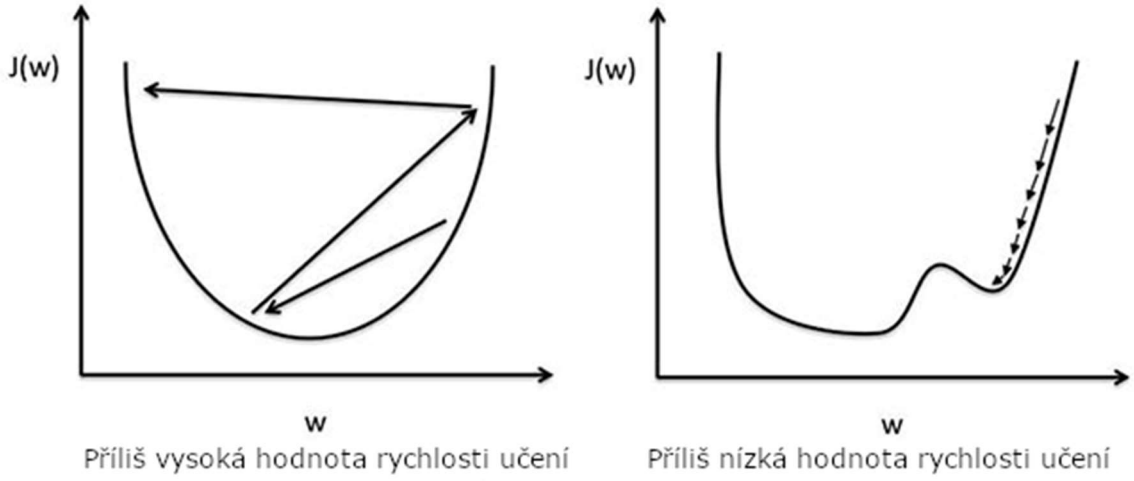
Rychlost učení

Velikost kroku neboli kvantum, o které změníme váhové či bias koeficienty při optimalizaci ML modelu, přímo ovlivňuje jak rychlost, tak přesnost GD. Tato velikost je vyjádřena parametrem, který se nazývá „rychlost učení“ (LeR) (z angl. learning rate). Konkrétně je LeR konfigurovatelným hyperparametrem používaným při tréninku neuronových sítí, který má malou kladnou hodnotu, často v rozmezí mezi 0 až 1. Menší hodnoty LeR vyžadují více iterací v průběhu tréninkového procese vzhledem k menším změnám hodnot při aktualizaci, zatímco vyšší hodnoty LeR vedou k rychlým změnám a vyžadují méně tréninkových interací. Jelikož tento parametr přímo ovlivňuje počet výpočetních iterací v tréninkovém procesu, tak jeho konfigurací jsme schopni výrazně změnit dobu trvání tohoto procesu. Existují však jistá omezení, které je nutno brát v úvahu.

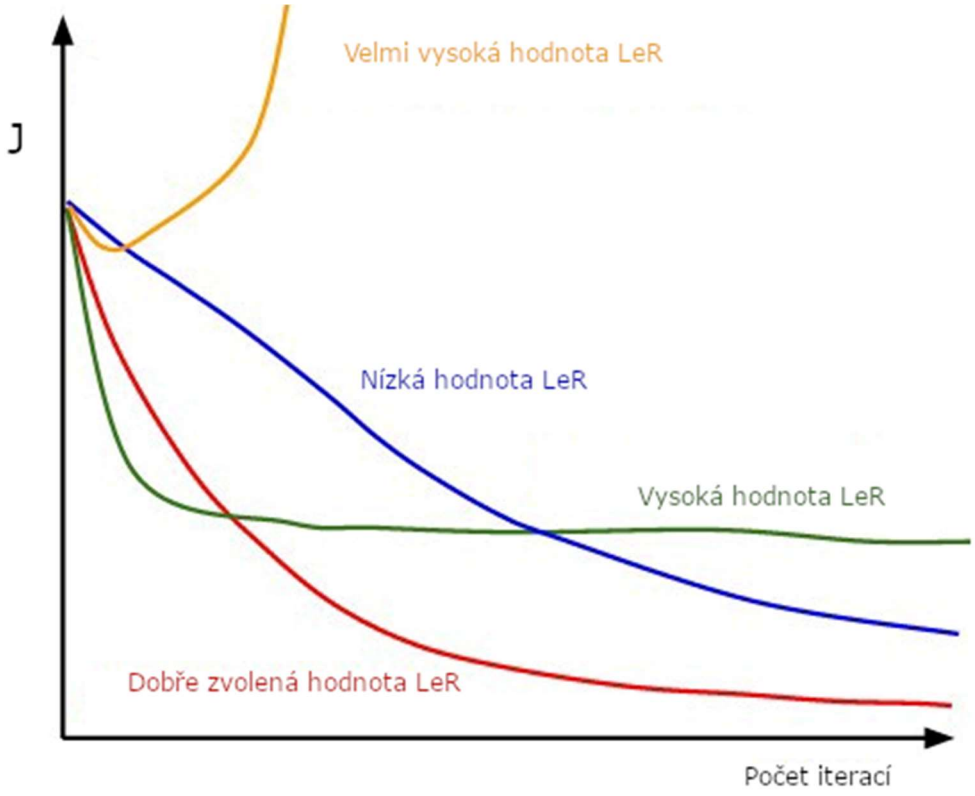
Příliš vysoká hodnota LeR může způsobit, že se GD bude příliš rychle pohybovat a nikdy nedosáhne hodnoty lokálního či globálního minima, jelikož daný konvexní úsek vždy „překročí“. V případě příliš nízké hodnoty LeR zase může nastat problém, že GD algoritmus uvízne v sedlovém bodě či neoptimálním lokálním minimu. Tuto problematiku znázorňuje obrázek 4.18.

Určení správné hodnoty LeR je experimentální záležitostí a správná hodnota se liší model od modelu. Je tedy potřeba nějakým způsobem monitorovat efektivitu tréninkového procesu pro různé experimentálně zvolené hodnoty LeR a následně pak zvolit hodnotu nejvhodnější. Již v předešlém textu bylo uvedeno, že čím více GD algoritmus spěje do lokálního (globálního) minima, tím více se snižuje hodnota CHF. Monitorace průběhu hodnot CHF pro jednotlivé iterace GD se proto jeví jako vhodný způsob pro evaluaci jednotlivých hodnot LeR. V ideálním případě má průběh hodnot CHF hyperbolický tvar, tedy hodnota CHF poměrně rychle klesá a následně se ustálí v důsledku dosažení lokálního minima. Různé příklady průběhů CHF jsou vidět na

obrázku 4.19. Při nastavení velmi nízké hodnoty LeR bude trénovací proces trvat velmi dlouhou dobu a CHF bude klesat pomalu. Naopak při velmi vysoké hodnotě nebude trénovací algoritmus schopen dosáhnout lokálního minima a CHF nikdy nedosáhne dostatečně nízkých hodnot, naopak v extrémním případě může začít nekontrolovatelně stoupat.



Obrázek 4.19: Problémy při nevhodně zvolených hodnotách LeR, převzato z [31], upraveno



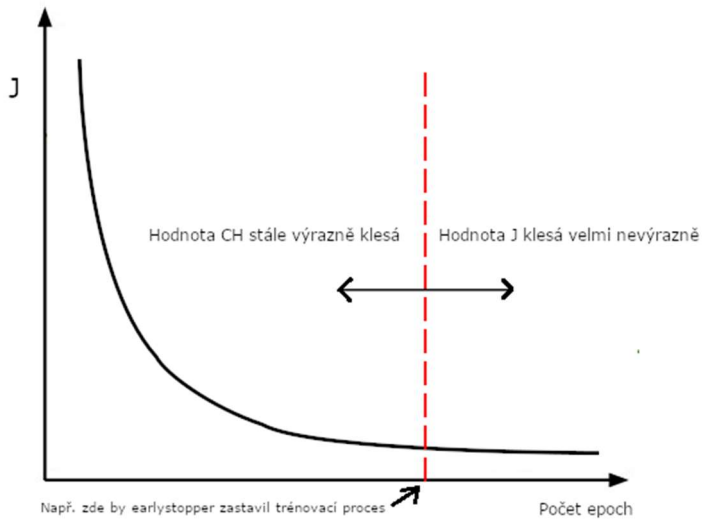
Obrázek 4.18: Průběhy CHF při různých hodnotách LeR, převzato z [32], upraveno

Velikost batch

Jak již bylo zmíněno výše, při implementaci MSGD jsou prováděny jednotlivé iterace na základně provedení výpočtů pro jednotlivé skupiny instancí, kterým se nazývají batche. Počet instancí v jednom batchi je dalším hyperparametrem, který programátor může sám modifikovat. Pokud bychom používali při aktualizaci hodnoty vypočtené na základě jednoho či více instancí, zvětšuje se tak riziko zanesení náhodné chyby a šumu. Je tedy vhodnější provádět výpočet na základě zprůměrování hodnot získaných z více instancí. Větší velikost batche však s sebou nese nevýhodu v podobě více paměti pro tréninkový proces. Je tedy nutné opět činit kompromis, přičemž doporučují se velikosti batche: 1, 2, 4, 8, 16, 32, 64, 128, 256. V praxi se jeví jako nejběžnější hodnota 32 a proto je také tato hodnota použita při vytváření modelů pro klasifikaci fibrilujících a nefibrilujících pacientů.

Počet epoch

Pojmem epocha se v kontextu ML myslí jeden tréninkový cyklus, kdy je ML model vystaven všem instancím dat z tréninkového datasetu. V praxi je však potřeba ML model vystavit trénovacím datům mnohem vícekrát nežli pouze jednou. Jako indikátor potřebného počtu opět poslouží monitorace průběhu CHF. Je potřeba pokračovat v trénovacím procesu do té doby, dokud má hodnota CHF tendenci klesat, viz obrázek 4.20.



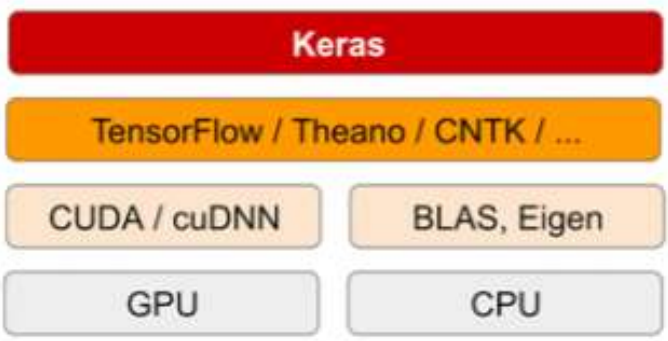
Obrázek 4.20: Počet epoch potřebný pro minimalizaci CHF, autor

Existují dvě možnosti, jak nastavit hodnotu počtu epoch, které při trénovacím procesu proběhnou. První možností je nastavit hodnotu počtu epoch ručně, kdy se okometricky zhodnotí graf průběhu CHF a dle uvážení se odečte z horizontální osy počet epoch. Druhým způsobem je poté využít funkce zvané „earlystopper“. Tato přídatná funkce funguje na principu porovnávání hodnot CHF pro jednotlivý sousední epochy. Lze

nastavit minimální možnou změnu CHF, kterou bude earlystopper tolerovat. Pokud hodnota změny bude menší nežli nastavená, trénovací proces se zastaví. Také je možno určit dobu tolerance určením počtu epoch, po které nemusí dojít k žádné změně. Tato metoda je implementována v použitých ML modelech.

4.6 Keras a Tensorflow

Keras je DL framework, který je napsán v programovacím jazyce Python a umožňuje navrhnout a následně i natrénovat téměř jakýkoliv typ DL modelu bez nutnosti explicitního programování veškerých částí modelu. Jedná se vlastně o knihovnu jednotlivých stavebních bloků modelu, které lze snadně aplikovat. Keras je tzv. „high-level“ framework, tudíž není programován k tzv. „low-level“ manipulaci s jednotlivými tensory. Výraz tensor lze chápat jako generalizovaná matice. Termín „hodnost“ (z angl. rank) poté vyjadřuje počet dimenzí daného tensoru. Např. tensor s hodností 0 lze chápat jako skalár, s hodností jedna jako vektor, s hodností 2 jako matici atd. V kontextu ML obecně lze počet dimenzí tensoru přirovnat k počtu vstupních parametrů v datasetu.



Obrázek 4.21: Modulové schéma Keras, převzato z [15]

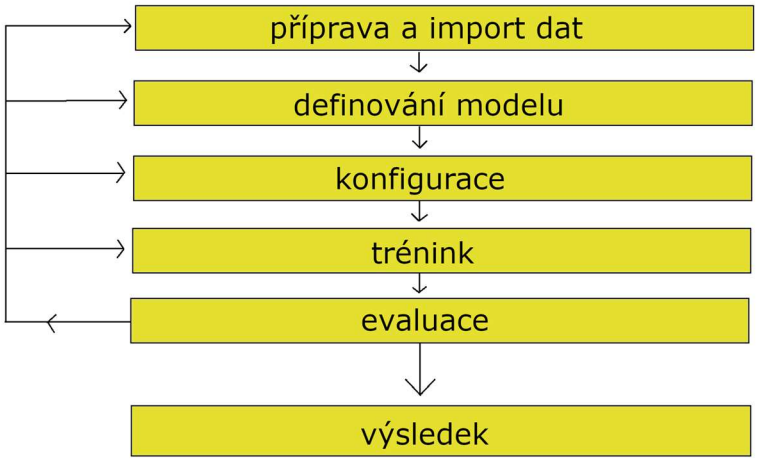
Keras se opírá hned o několik tzv. „backend enginů“, což jsou softwarové balíčky, které jsou programovány a velmi dobře optimalizovány pro účely low-level výpočetních procesů s tensory. V současné době existuje několik těchto backend enginů, přičemž třemi nejvíce používanými jsou Tensorflow vyvíjený společností Google, Microsoft Cognitive Toolkit (CNTK) vyvíjený společností Microsoft a Theano, který je vyvíjen MILA výzkumným centrem Montrealské Univerzity. V této práci bude použit Tensorflow, který je nejvíce rozšířený ze tří jmenovaných backend enginů. Pro programování v Pythonu použito vývojové prostředí Spyder 3.

Prostřednictvím TensorFlow (nebo Theano nebo CNTK) je Keras schopen bezproblémově běžet jak na procesorové jednotce (CPU), tak i na jednotce grafické karty (GPU). V případě použití CPU využívá knihovny BLAS či Eigen pro low-level

manipulaci, v případě GPU pak knihovnu Deep Neural Network NVIDIA CUDA (cuDNN).

4.6.1 Proces navrhování ML modelů v Keras

Proces navrhování ML modelu v Keras by se dal shrnout do několika po sobě následujících kroků viz obrázek 4.22.



Obrázek 4.22: Schéma procesu konstrukce ML modelu v Keras, autor

Prvním krokem je importace dat a jejich následná příprava, čímž se myslí rozdělení datasetu na tréninkovou a testovací skupinu. Většinou se 80 % dat použije na trénování modelu a zbylých 20 % pro jeho následné testování. Z důvodu reprezentativnosti je rozdělení určeno zcela náhodně.

Po přípravě dat následuje definování modelu. Je třeba určit typ modelu na základě vhodnosti vzhledem k použitému typu dat. Dále je pak třeba navrhnout jeho architekturu (počet perceptronů v jednotlivých vrstvách, počet vrstev). Model lze definovat dvěma způsoby, a to pomocí tzv. „sequential class“ či pomocí „functional API“ [15]. Sequantial class se používá v drtivé většině případů, ML model sestává z lineárních vrstev, příkladem tohoto typu architektury může být například obrázek 4.9. Functional API se používá k budování naprosto libovolných typů uspořádání jednotlivých perceptronů v rámci ML modelu a používá se jen zřídka. V této práci bude model sestavován pomocí sequential class, ukázka definice modelu z použitého kódu je na obrázku 4.23.

```
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
```

Obrázek 4.23: Definování modelu v Keras, autor

Jakmile je definována architektura ML modelu, je potřeba model konfigurovat. Toto se provádí pomocí kompilačního submodulu. V tomto kroku se určí typ CHF, LeR, metoda sledování přesnosti modelu a další parametry. Také lze do modelu implementovat funkci earlystopper. Ukázka z kódu je vidět na obrázku 4.24.

```
from keras.optimizers import Adam
from keras.callbacks import EarlyStopping

model.compile(Adam(learning_rate=0.001, beta_1=0.9, beta_2=0.999, amsgrad=False),
              loss = 'binary_crossentropy',
              metrics = ['accuracy'])

earlyStopper = EarlyStopping(monitor = 'val_loss',
                             min_delta = 0.003,
                             patience = 10,
                             verbose = 0,
                             mode = 'auto')
```

Obrázek 4.24: Konfigurace modelu v Keras, autor

V této fázi je struktura modelu kompletní, tudíž je třeba začít s tréninkovým procesem pomocí „fit“ funkce a následnou evaluací výsledků modelu, viz obrázek 4.25. V případě nesuficientních výsledků se lze po evaluaci zpětně vrátit k jednotlivým předešlým krokům a pozměnit jednotlivé parametry či strukturu dat a modelu.

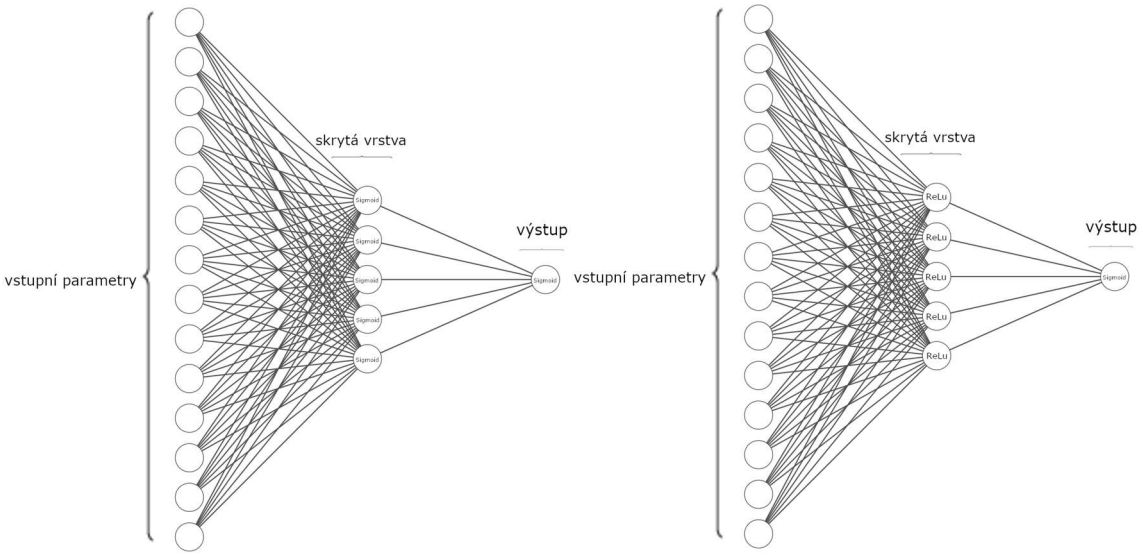
```
model.fit(xTrain, yTrain, batch_size = 32,
          epochs = 1000,
          callbacks = [earlyStopper],
          validation_split = 0.1,
          verbose = 0)

score = model.evaluate(xTest,yTest, batch_size = 32)
yPred = model.predict(xTest)
comparison = np.column_stack((yTest, yPred))
```

Obrázek 4.25: Trénování modelu a evaluace výsledků v Keras, autor

4.7 Navrhnuté ML modely pro klasifikaci

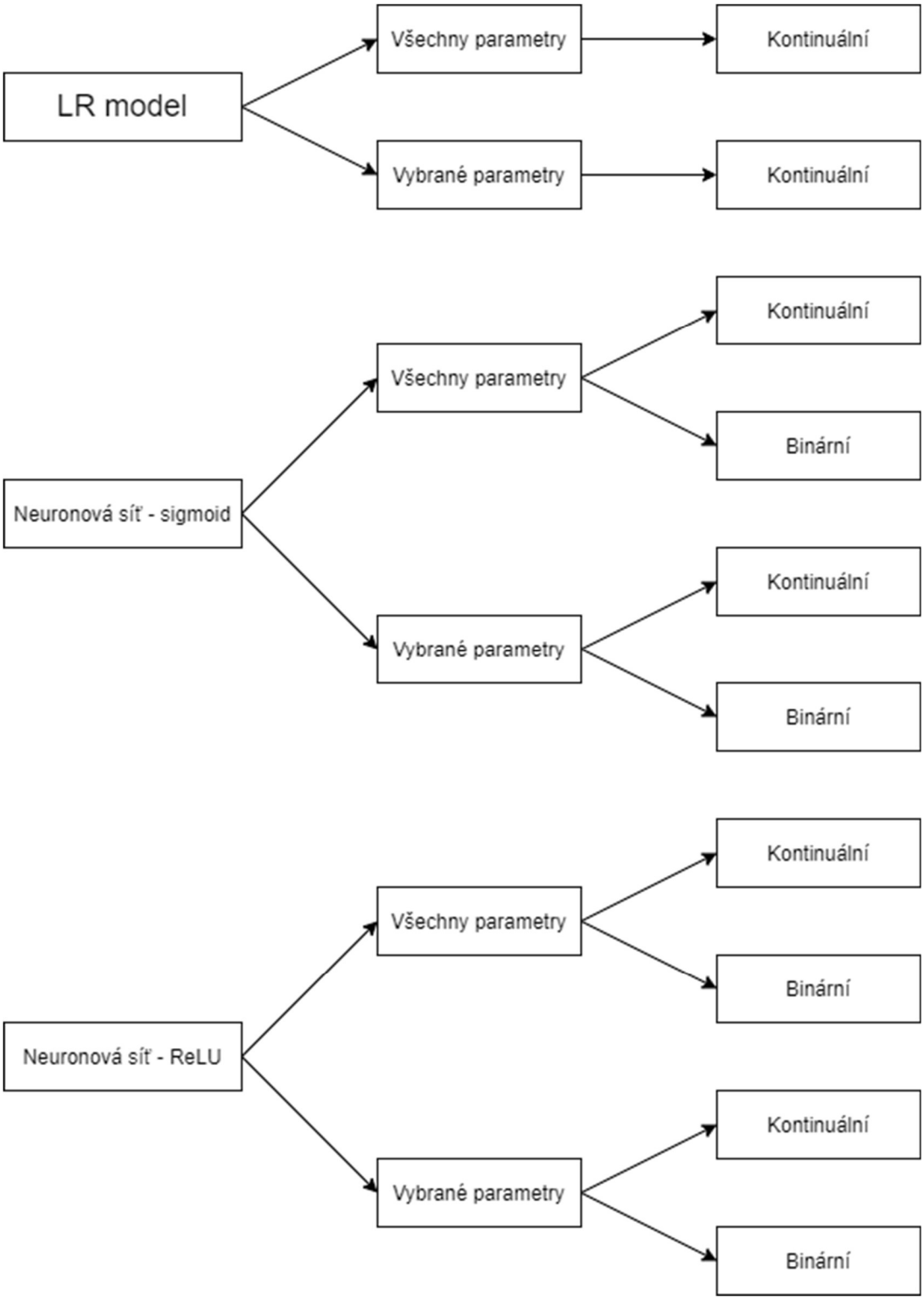
Celkově byly pro klasifikaci použity 3 ML modely: LR model a dvě neuronové sítě lišící se aktivační funkcí ve skryté vrstvě neuronů – funkce sigmoid a RELU, viz. obrázek 4.26. Dále byly pak zvoleny dva sety vstupních parametrů, se kterými ML model pracoval. V prvním setu bylo použito všech parametrů, u kterých bylo prokázáno odlišné rozdělení fibrilující a nefibrilující skupiny pomocí Mann-Whitneyho U testu. Druhý set pak sestával pouze z parametrů, pro které byla hodnota p menší či rovna 0.001. Přehled parametrů pro každý set je uveden v tabulce 5.3. Tyto sety měly jak kontinuální, tak binární reprezentaci. Neuronové sítě byly implementovaný na oboje tyto varianty, LR model byl

implementován pouze na kontinuální set. Celkové schéma všech navržených modelů je vidět na obrázku 4.27.



Obrázek 4.26: Navrhnuté neuronové sítě s aktivační funkcí sigmoid vlevo, ReLU vpravo, vytvořeno pmocí http://alexlenail.me/NN-SVG/LeNet.html

Pro každý model proběhlo celkem 100 iterací (z důvodu statistické reprezentativnosti), a to pro několik hodnot thresholdu. Z důvodu značné nevyrovnanosti počtu pacientů v jednotlivých třídách bylo při každé iteraci náhodně vybráno 30 pacientů, kteří nefibrilovali. Nerovnoměrnost mezi jednotlivými třídami v rámci trénovacích dat může mít zásadní vliv na výkonnost ML modelu, jelikož bylo prokázáno, že v některých případech způsobuje významné omezení výkonu dosažitelného standardními ML metodami, které předpokládají vyvážené rozdělení třídy [33]. Jednotlivé confusion matice vygenerované při každé iteraci byly sečteny a následně pak na jejich základě byly vypočítány průměrné hodnoty senzitivity, specificity a preciznosti pro konkrétní model. Jako parametr pro evaluaci celkového výkonu ML modelu pro jednotlivé hodnoty thresholdů byl vybrán součin specificity a senzitivity. Čím vyšší je tato hodnota, tím lepší je celková výkonnost ML modely. Tímto se eliminují vysoké hodnoty jednoho parametru na úkor velmi nízké hodnoty druhého parametru. V tabulce 5.14 se poté nachází seznam maximálních hodnot těchto součinů pro jednotlivé ML modely a hodnoty thresholdů. V případě více stejných maximálních hodnot je uvedena taková hodnota, při které je vyšší senzitivita.



Obrázek 4.27: Schéma všech navržených metod a vstupních dat, vytvořeno pomocí https://app.diagrams.net/

5 Výsledky

5.1 Předzpracování datasetu

Původní dataset obsahoval data 583 pacientů, z nichž 41 prodělalo VF a 542 neprodělalo VF. Po eliminaci pacientů s neúplným záznam či chybnými hodnotami parametrů nakonec výsledný dataset sestával ze 408 pacientských záznamů, přičemž 23 pacientů prodělalo VF a 385 pacientů neprodělalo VF. V tabulce 5.1 jsou pacienti, kteří neprodělali VF označeni jako „nVF“ a ti co prodělali jako „VF“.

Tabulka 5.1: Složení datasetu před a po zpracování

	Počet pacientů		
	nVF	VF	celkový
Původní dataset	542	41	583
Zpracovaný dataset	385	23	408

5.2 Mann – Whitney U test

Výsledky Mann – Whitneyho U testu pro testování odlišného rozdělení nVF a VF skupiny jsou pro každý parametr uvedeny v tabulce 5.2 níže. Hladina významnosti α = 5 %. Nulová hypotéza H_0 v tomto případě potvrzuje odlišnou distribuci parametrů mezi oběma skupinami. Mezní hladina významnosti (*p* hodnota) byla 0.05. U jednotlivých skupin je uveden medián a interval IQR.

Tabulka 5.2: Výsledky Mann – Whitney U testu pro odlišné rozdělení nVF a VF skupiny pro jednotlivé parametry

Parametr	nVF skupina n = 385		VF skupina n = 23		*p* hodnota	Výsledek H_0
	Medián	IQR	Medián	IQR		
Tamp_max (mV)	523.1	[374.4;728.6]	716.8	[532.8, 1070.0]	<0.001	platí
Tamp_min (mV)	-236.1	[-328.2;-156.1]	-381.6	[-568.2, -276.6]	<0.001	platí
Tamp_disp (mV)	777.4	[586.3;1060.7]	1144.6	[868.4, 1626.4]	<0.001	platí
G_TpkTend (ms)	128.0	[113.0;148.0]	140.0	[125.0, 150.0]	0.014	platí
summaST (mV)	1506.5	[970.5;2158.5]	2538.0	[1800.0, 3699.0]	<0.001	platí
max_ST (mV)	290.5	[179.0;452.2]	483.5	[300.0, 695.0]	<0.001	platí
OverallQRS (ms)	98.0	[88.0;108.0]	104.0	[94.0, 124.0]	0.019	platí
HR (bpm)	75.0	[65.0;90.0]	78.0	[63.8, 86.8]	0.958	zamítnuta
Tarea_V1 (ms·V)	606.0	[-202.5;1428.5]	-212.5	[-1143.5, 1252.5]	0.076	zamítnuta
Tarea_V2 (ms·V)	2054.5	[122.2;4197.2]	492.0	[-2086.8, 2749.8]	0.045	platí
Tarea_V3 (ms·V)	2017.5	[494.2;4304.0]	2579.0	[213.8, 6096.2]	0.659	zamítnuta
Tarea_V4 (ms·V)	1553.5	[134.5;3225.8]	2137.0	[495.5, 5628.0]	0.164	zamítnuta
Tarea_V5 (ms·V)	1093.5	[-58.8;2191.8]	1630.0	[958.8, 3803.8]	0.028	platí
Tarea_V6 (ms·V)	710.5	[-82.0;1561.2]	1261.0	[657.0, 2832.2]	0.012	platí
Tarea_I (ms·V)	157.5	[-568.8;1006.2]	-284.5	[-1176.5, 809.2]	0.171	zamítnuta
Tarea_II (ms·V)	1269.5	[437.2;2473.2]	3130.0	[1701.5, 4454.0]	<0.001	platí
Tarea_III (ms·V)	1004.0	[-96.8;2373.2]	3629.5	[1197.0, 5798.2]	<0.001	platí
Tarea_aVL (ms·V)	-853.0	[-1548.0;272.0]	-1495.5	[-2248.2, -821.5]	0.001	platí
Tarea_aVR (ms·V)	-464.5	[-1457.0;436.8]	-2543.0	[-3498.5, -160.5]	0.001	platí
Tarea_aVF (ms·V)	1030.5	[208.2;2525.0]	3497.0	[1650.2, 4726.5]	<0.001	platí

Z výsledků z tabulky 5.2 je vidět, že u parametrů s rozdílnou hodnotou mediánu pro nVF a VF skupin společně s relativně malým rozpětím IQR bylo většinou potvrzeno rozdílné rozdělení těchto dvou skupin, resp. nulová hypotéza Mann – Whitney U testu nebyla vyvrácena. Mediány všech parametrů kromě Tarea_V2 nabývaly větších hodnot pro VF skupinu. Následující tabulka 5.3 poté obsahuje seznam všech parametrů, u kterých bylo prokázáno rozdílné rozdělení, a to jak pro všechny hodnoty p (set 1), tak pouze pro hodnoty $p <= 0{,}001$ (set 2).

Tabulka 5.3: Přehled parametrů v jednotlivých setech

Set parametrů 1 - (H_0 platí)	Set parametrů 2 - (H_0 platí, $p <= 0.001$)
Tamp_max	Tamp_max
Tamp_min	Tamp_min
Tamp_disp	Tamp_disp
G_TpkTend	summaST
summaST	max_ST
max_ST	Tarea_II
OverallQRS	Tarea_III
Tarea_V2	Tarea_aVL
Tarea_V5	Tarea_aVR
Tarea_V6	Tarea_aVF
Tarea_II	
Tarea_III	
Tarea_aVL	
Tarea_aVR	
Tarea_aVF	

V následujících tabulkách 5.4 – 5.13 jsou uvedeny výsledky klasifikace pro jednotlivé navržené ML modely, viz. obrázek 4.27. Tabulkách 5.5 a 5.6 jsou uvedeny výsledky pro ML model logistické regrese, v tabulkách 5.7 – 5.13 jsou uvedeny výsledky neuronových sítí. V každé z tabulek 5.5 – 5.13 je vždy zvýrazněn jeden řádek, který odpovídá maximální hodnotě součinu specificity a sensitivity v rámci konkrétního ML modelu. Výčet těchto maximálních hodnot součinů společně s názvy modelů a hodnotami thresholdů je uveden v tabulce 5.14.

Při porovnání výsledků LR a neuronových sítí lze konstatovat, že neuronové sítě se celkově vyznačovaly vyšší výkonností. Jejich celková přesnost je zpravidla větší, než při použití LR a také tolik nekolísá při volbách rozdílných hodnot thresholdů. Naopak u LR hodnota přesnosti více kolísá a nejmenší je pro střední hodnoty thresholdů, naopak největší je pro největší hodnoty. Zatímco poměr sensitivity a specificity u LR modelu je ve značné disbalanci, neuronové sítě se vyznačují, zvláště pak pro střední hodnot thresholdů, poměrně vyrovnaným poměrem těchto dvou metrických parametrů.

Tabulka 5.4 – Výsledky logistické regrese

Použitá data:	Set parametrů 1				
Threshold (%)	Confusion matice (-)	Přesnost (%)	Sensitivita (%)	Specificita (%)	Preciznost (%)
5	$\begin{vmatrix} 10 & 629 \\ 63 & 398 \end{vmatrix}$	37	86	2	39
10	$\begin{vmatrix} 36 & 608 \\ 103 & 353 \end{vmatrix}$	35	77	6	37
15	$\begin{vmatrix} 47 & 573 \\ 131 & 349 \end{vmatrix}$	36	73	8	38
20	$\begin{vmatrix} 58 & 568 \\ 169 & 305 \end{vmatrix}$	33	64	9	35
25	$\begin{vmatrix} 70 & 580 \\ 158 & 292 \end{vmatrix}$	33	65	11	33
30	$\begin{vmatrix} 97 & 526 \\ 221 & 256 \end{vmatrix}$	32	54	16	33
35	$\begin{vmatrix} 109 & 531 \\ 214 & 246 \end{vmatrix}$	32	53	17	32
40	$\begin{vmatrix} 144 & 495 \\ 231 & 230 \end{vmatrix}$	34	50	23	32
45	$\begin{vmatrix} 146 & 489 \\ 266 & 199 \end{vmatrix}$	31	43	23	29
50	$\begin{vmatrix} 177 & 444 \\ 277 & 202 \end{vmatrix}$	34	42	29	31
55	$\begin{vmatrix} 212 & 416 \\ 294 & 178 \end{vmatrix}$	35	38	34	30
60	$\begin{vmatrix} 240 & 364 \\ 337 & 159 \end{vmatrix}$	36	32	40	30
65	$\begin{vmatrix} 274 & 367 \\ 340 & 119 \end{vmatrix}$	36	26	43	24
70	$\begin{vmatrix} 314 & 304 \\ 386 & 96 \end{vmatrix}$	37	20	51	24
75	$\begin{vmatrix} 382 & 214 \\ 424 & 80 \end{vmatrix}$	42	16	64	27
80	$\begin{vmatrix} 449 & 158 \\ 431 & 62 \end{vmatrix}$	46	13	74	28
85	$\begin{vmatrix} 518 & 111 \\ 440 & 31 \end{vmatrix}$	50	7	82	22
90	$\begin{vmatrix} 555 & 51 \\ 475 & 19 \end{vmatrix}$	52	4	92	27
95	$\begin{vmatrix} 622 & 20 \\ 464 & 17 \end{vmatrix}$	57	4	97	46

Tabulka 5.5 Výsledky logistické regrese

Použitá data:	Set parametrů 2				
Threshold (%)	Confusion matice (-)	Přesnost (%)	Sensitivita (%)	Specificita (%)	Preciznost (%)
5	$\begin{vmatrix} 20 & 599 \\ 74 & 407 \end{vmatrix}$	39	85	3	40
10	$\begin{vmatrix} 24 & 573 \\ 108 & 395 \end{vmatrix}$	38	79	4	41
15	$\begin{vmatrix} 34 & 584 \\ 117 & 365 \end{vmatrix}$	36	76	6	38
20	$\begin{vmatrix} 45 & 562 \\ 143 & 350 \end{vmatrix}$	36	71	7	38
25	$\begin{vmatrix} 49 & 553 \\ 158 & 340 \end{vmatrix}$	35	68	8	38
30	$\begin{vmatrix} 55 & 559 \\ 186 & 300 \end{vmatrix}$	32	62	9	35
35	$\begin{vmatrix} 94 & 533 \\ 238 & 235 \end{vmatrix}$	30	50	15	31
40	$\begin{vmatrix} 119 & 525 \\ 239 & 217 \end{vmatrix}$	31	48	18	29
45	$\begin{vmatrix} 121 & 482 \\ 258 & 239 \end{vmatrix}$	33	48	20	33
50	$\begin{vmatrix} 156 & 465 \\ 282 & 197 \end{vmatrix}$	32	41	25	30
55	$\begin{vmatrix} 198 & 435 \\ 287 & 180 \end{vmatrix}$	34	39	31	29
60	$\begin{vmatrix} 233 & 385 \\ 338 & 144 \end{vmatrix}$	34	30	38	27
65	$\begin{vmatrix} 299 & 319 \\ 337 & 145 \end{vmatrix}$	40	30	48	31
70	$\begin{vmatrix} 381 & 231 \\ 403 & 85 \end{vmatrix}$	42	17	62	27
75	$\begin{vmatrix} 427 & 175 \\ 434 & 64 \end{vmatrix}$	45	13	71	27
80	$\begin{vmatrix} 504 & 116 \\ 445 & 35 \end{vmatrix}$	49	7	81	23
85	$\begin{vmatrix} 568 & 57 \\ 449 & 26 \end{vmatrix}$	54	5	91	31
90	$\begin{vmatrix} 611 & 16 \\ 467 & 6 \end{vmatrix}$	56	1	97	27
95	$\begin{vmatrix} 618 & 3 \\ 474 & 5 \end{vmatrix}$	57	1	100	63

Tabulka 5.6 – Výsledky neuronové sítě

Typ aktivační funkce:		Sigmoid			
Použitá data:		Set parametrů 1, kontinuální			
Threshold (%)	Confusion matice (-)	Přesnost (%)	Sensitivita (%)	Specificita (%)	Preciznost (%)
5	$\begin{vmatrix} 3 & 639 \\ 4 & 454 \end{vmatrix}$	42	99	0	42
10	$\begin{vmatrix} 68 & 550 \\ 31 & 451 \end{vmatrix}$	47	94	11	45
15	$\begin{vmatrix} 169 & 463 \\ 67 & 401 \end{vmatrix}$	52	86	27	46
20	$\begin{vmatrix} 238 & 356 \\ 104 & 402 \end{vmatrix}$	58	79	40	53
25	$\begin{vmatrix} 332 & 305 \\ 110 & 353 \end{vmatrix}$	62	76	52	54
30	$\begin{vmatrix} 338 & 263 \\ 144 & 355 \end{vmatrix}$	63	71	56	57
35	$\begin{vmatrix} 385 & 244 \\ 152 & 319 \end{vmatrix}$	64	68	61	57
40	$\begin{vmatrix} 422 & 214 \\ 166 & 298 \end{vmatrix}$	65	64	66	58
45	$\begin{vmatrix} 464 & 181 \\ 194 & 261 \end{vmatrix}$	66	57	72	59
50	$\begin{vmatrix} 466 & 172 \\ 213 & 249 \end{vmatrix}$	65	54	73	59
55	$\begin{vmatrix} 503 & 157 \\ 210 & 230 \end{vmatrix}$	67	52	76	59
60	$\begin{vmatrix} 495 & 108 \\ 244 & 253 \end{vmatrix}$	68	51	82	70
65	$\begin{vmatrix} 527 & 101 \\ 257 & 215 \end{vmatrix}$	67	46	84	68
70	$\begin{vmatrix} 524 & 67 \\ 297 & 212 \end{vmatrix}$	67	42	89	76
75	$\begin{vmatrix} 565 & 71 \\ 293 & 171 \end{vmatrix}$	67	37	89	71
80	$\begin{vmatrix} 575 & 67 \\ 299 & 159 \end{vmatrix}$	67	35	90	70
85	$\begin{vmatrix} 562 & 45 \\ 380 & 113 \end{vmatrix}$	61	23	93	72
90	$\begin{vmatrix} 602 & 28 \\ 405 & 65 \end{vmatrix}$	61	14	96	70
95	$\begin{vmatrix} 613 & 4 \\ 471 & 12 \end{vmatrix}$	57	2	99	75

Tabulka 5.7– Výsledky neuronové sítě

Typ aktivační funkce:		Sigmoid			
Použitá data:		Set parametrů 1, binární			
Threshold (%)	Confusion matice (-)	Přesnost (%)	Sensitivita (%)	Specificita (%)	Preciznost (%)
5	$\begin{vmatrix} 154 & 477 \\ 30 & 439 \end{vmatrix}$	54	94	24	48
10	$\begin{vmatrix} 271 & 355 \\ 55 & 419 \end{vmatrix}$	63	88	43	54
15	$\begin{vmatrix} 315 & 305 \\ 80 & 400 \end{vmatrix}$	65	83	51	57
20	$\begin{vmatrix} 334 & 292 \\ 91 & 383 \end{vmatrix}$	65	81	53	57
25	$\begin{vmatrix} 389 & 223 \\ 109 & 379 \end{vmatrix}$	70	78	64	63
30	$\begin{vmatrix} 385 & 224 \\ 118 & 373 \end{vmatrix}$	69	76	63	62
35	$\begin{vmatrix} 417 & 227 \\ 98 & 358 \end{vmatrix}$	70	79	65	61
40	$\begin{vmatrix} 423 & 197 \\ 133 & 347 \end{vmatrix}$	70	72	68	64
45	$\begin{vmatrix} 464 & 175 \\ 141 & 320 \end{vmatrix}$	71	69	73	65
50	$\begin{vmatrix} 458 & 187 \\ 158 & 297 \end{vmatrix}$	69	65	71	61
55	$\begin{vmatrix} 475 & 154 \\ 172 & 299 \end{vmatrix}$	70	63	76	66
60	$\begin{vmatrix} 473 & 134 \\ 188 & 305 \end{vmatrix}$	71	62	78	69
65	$\begin{vmatrix} 516 & 114 \\ 212 & 258 \end{vmatrix}$	70	55	82	69
70	$\begin{vmatrix} 521 & 71 \\ 228 & 280 \end{vmatrix}$	73	55	88	80
75	$\begin{vmatrix} 520 & 77 \\ 234 & 269 \end{vmatrix}$	72	53	87	78
80	$\begin{vmatrix} 582 & 56 \\ 273 & 189 \end{vmatrix}$	70	41	91	77
85	$\begin{vmatrix} 579 & 58 \\ 282 & 181 \end{vmatrix}$	69	39	91	76
90	$\begin{vmatrix} 600 & 27 \\ 371 & 102 \end{vmatrix}$	64	22	96	79
95	$\begin{vmatrix} 619 & 7 \\ 447 & 27 \end{vmatrix}$	59	6	99	79

Tabulka 5.8– Výsledky neuronové sítě

Typ aktivační funkce:		Sigmoid			
Použitá data:		Set parametrů 2, kontinuální			
Threshold (%)	Confusion matice (-)	Přesnost (%)	Sensitivita (%)	Specificita (%)	Preciznost (%)
5	$\begin{vmatrix} 0 & 635 \\ 0 & 465 \end{vmatrix}$	42	100	0	42
10	$\begin{vmatrix} 0 & 634 \\ 0 & 466 \end{vmatrix}$	42	100	0	42
15	$\begin{vmatrix} 10 & 596 \\ 5 & 489 \end{vmatrix}$	45	99	2	45
20	$\begin{vmatrix} 36 & 602 \\ 11 & 451 \end{vmatrix}$	44	98	6	43
25	$\begin{vmatrix} 115 & 514 \\ 35 & 438 \end{vmatrix}$	50	93	18	46
30	$\begin{vmatrix} 214 & 415 \\ 68 & 403 \end{vmatrix}$	56	86	34	49
35	$\begin{vmatrix} 339 & 261 \\ 120 & 380 \end{vmatrix}$	65	76	57	59
40	$\begin{vmatrix} 394 & 218 \\ 164 & 324 \end{vmatrix}$	65	66	64	60
45	$\begin{vmatrix} 457 & 169 \\ 178 & 296 \end{vmatrix}$	68	62	73	64
50	$\begin{vmatrix} 502 & 126 \\ 209 & 263 \end{vmatrix}$	70	56	80	68
55	$\begin{vmatrix} 539 & 92 \\ 267 & 202 \end{vmatrix}$	67	43	85	69
60	$\begin{vmatrix} 528 & 77 \\ 320 & 175 \end{vmatrix}$	64	35	87	69
65	$\begin{vmatrix} 579 & 42 \\ 354 & 125 \end{vmatrix}$	64	26	93	75
70	$\begin{vmatrix} 554 & 34 \\ 408 & 104 \end{vmatrix}$	60	20	94	75
75	$\begin{vmatrix} 626 & 11 \\ 412 & 51 \end{vmatrix}$	62	11	98	82
80	$\begin{vmatrix} 610 & 3 \\ 473 & 14 \end{vmatrix}$	57	3	100	82
85	$\begin{vmatrix} 610 & 12 \\ 495 & 16 \end{vmatrix}$	55	3	98	57
90	$\begin{vmatrix} 621 & 0 \\ 479 & 0 \end{vmatrix}$	56	0	100	-
95	$\begin{vmatrix} 664 & 0 \\ 436 & 0 \end{vmatrix}$	60	0	100	-

Tabulka 5.9– Výsledky neuronové sítě

Typ aktivační funkce:		Sigmoid			
Použitá data:		Set parametrů 2, binární			
Threshold (%)	Confusion matice (-)	Přesnost (%)	Sensitivita (%)	Specificita (%)	Preciznost (%)
5	$\begin{vmatrix} 70 & 558 \\ 3 & 469 \end{vmatrix}$	49	99	11	46
10	$\begin{vmatrix} 185 & 448 \\ 40 & 427 \end{vmatrix}$	56	91	29	49
15	$\begin{vmatrix} 287 & 359 \\ 54 & 400 \end{vmatrix}$	62	88	44	53
20	$\begin{vmatrix} 306 & 299 \\ 101 & 394 \end{vmatrix}$	64	80	51	57
25	$\begin{vmatrix} 329 & 284 \\ 110 & 377 \end{vmatrix}$	64	77	54	57
30	$\begin{vmatrix} 359 & 272 \\ 118 & 351 \end{vmatrix}$	65	75	57	56
35	$\begin{vmatrix} 387 & 214 \\ 140 & 359 \end{vmatrix}$	68	72	64	63
40	$\begin{vmatrix} 424 & 219 \\ 133 & 324 \end{vmatrix}$	68	71	66	60
45	$\begin{vmatrix} 448 & 160 \\ 160 & 332 \end{vmatrix}$	71	67	74	67
50	$\begin{vmatrix} 456 & 148 \\ 176 & 318 \end{vmatrix}$	70	64	75	68
55	$\begin{vmatrix} 493 & 126 \\ 192 & 289 \end{vmatrix}$	71	60	80	70
60	$\begin{vmatrix} 506 & 130 \\ 186 & 278 \end{vmatrix}$	71	60	80	68
65	$\begin{vmatrix} 508 & 100 \\ 203 & 289 \end{vmatrix}$	72	59	84	74
70	$\begin{vmatrix} 544 & 90 \\ 227 & 239 \end{vmatrix}$	71	51	86	73
75	$\begin{vmatrix} 562 & 82 \\ 212 & 244 \end{vmatrix}$	73	54	87	75
80	$\begin{vmatrix} 570 & 44 \\ 298 & 188 \end{vmatrix}$	69	39	93	81
85	$\begin{vmatrix} 564 & 38 \\ 377 & 121 \end{vmatrix}$	62	24	94	76
90	$\begin{vmatrix} 589 & 18 \\ 459 & 34 \end{vmatrix}$	57	7	97	65
95	$\begin{vmatrix} 613 & 0 \\ 485 & 2 \end{vmatrix}$	56	0	100	100

Tabulka 5.10– Výsledky neuronové sítě

Typ aktivační funkce:		ReLU			
Použitá data:		Set parametrů 1, kontinuální			
Threshold (%)	Confusion matice (-)	Přesnost (%)	Sensitivita (%)	Specificita (%)	Preciznost (%)
5	$\begin{vmatrix} 208 & 409 \\ 72 & 411 \end{vmatrix}$	56	85	34	50
10	$\begin{vmatrix} 268 & 371 \\ 99 & 362 \end{vmatrix}$	57	79	42	49
15	$\begin{vmatrix} 291 & 342 \\ 98 & 369 \end{vmatrix}$	60	79	46	52
20	$\begin{vmatrix} 299 & 325 \\ 114 & 362 \end{vmatrix}$	60	76	48	53
25	$\begin{vmatrix} 376 & 249 \\ 129 & 346 \end{vmatrix}$	66	73	60	58
30	$\begin{vmatrix} 392 & 219 \\ 161 & 328 \end{vmatrix}$	65	67	64	60
35	$\begin{vmatrix} 425 & 171 \\ 183 & 321 \end{vmatrix}$	68	64	71	65
40	$\begin{vmatrix} 406 & 210 \\ 195 & 289 \end{vmatrix}$	63	60	66	58
45	$\begin{vmatrix} 431 & 190 \\ 211 & 268 \end{vmatrix}$	64	56	69	59
50	$\begin{vmatrix} 488 & 167 \\ 188 & 257 \end{vmatrix}$	68	58	75	61
55	$\begin{vmatrix} 470 & 166 \\ 200 & 264 \end{vmatrix}$	67	57	74	61
60	$\begin{vmatrix} 497 & 128 \\ 229 & 246 \end{vmatrix}$	68	52	80	66
65	$\begin{vmatrix} 487 & 105 \\ 259 & 249 \end{vmatrix}$	67	49	82	70
70	$\begin{vmatrix} 528 & 114 \\ 228 & 230 \end{vmatrix}$	69	50	82	67
75	$\begin{vmatrix} 543 & 107 \\ 231 & 219 \end{vmatrix}$	69	49	84	67
80	$\begin{vmatrix} 534 & 83 \\ 294 & 189 \end{vmatrix}$	66	39	87	69
85	$\begin{vmatrix} 561 & 80 \\ 268 & 191 \end{vmatrix}$	68	42	88	70
90	$\begin{vmatrix} 524 & 76 \\ 307 & 193 \end{vmatrix}$	65	39	87	72
95	$\begin{vmatrix} 568 & 43 \\ 341 & 148 \end{vmatrix}$	65	30	93	77

Tabulka 5.11– Výsledky neuronové sítě

Typ aktivační funkce:		ReLU			
Použitá data:		Set parametrů 1, binární			
Threshold (%)	Confusion matice (-)	Přesnost (%)	Sensitivita (%)	Specificita (%)	Preciznost (%)
5	$\begin{vmatrix} 294 & 329 \\ 73 & 404 \end{vmatrix}$	63	85	47	55
10	$\begin{vmatrix} 312 & 308 \\ 75 & 405 \end{vmatrix}$	65	84	50	57
15	$\begin{vmatrix} 357 & 271 \\ 88 & 384 \end{vmatrix}$	67	81	57	59
20	$\begin{vmatrix} 372 & 239 \\ 97 & 392 \end{vmatrix}$	69	80	61	62
25	$\begin{vmatrix} 401 & 240 \\ 114 & 345 \end{vmatrix}$	68	75	63	59
30	$\begin{vmatrix} 410 & 203 \\ 125 & 362 \end{vmatrix}$	70	74	67	64
35	$\begin{vmatrix} 419 & 188 \\ 128 & 365 \end{vmatrix}$	71	74	69	66
40	$\begin{vmatrix} 434 & 187 \\ 151 & 328 \end{vmatrix}$	69	68	70	64
45	$\begin{vmatrix} 445 & 181 \\ 148 & 326 \end{vmatrix}$	70	69	71	64
50	$\begin{vmatrix} 459 & 165 \\ 165 & 311 \end{vmatrix}$	70	65	74	65
55	$\begin{vmatrix} 471 & 148 \\ 164 & 317 \end{vmatrix}$	72	66	76	68
60	$\begin{vmatrix} 498 & 125 \\ 184 & 293 \end{vmatrix}$	72	61	80	70
65	$\begin{vmatrix} 525 & 125 \\ 159 & 291 \end{vmatrix}$	74	65	81	70
70	$\begin{vmatrix} 526 & 106 \\ 217 & 251 \end{vmatrix}$	71	54	83	70
75	$\begin{vmatrix} 520 & 107 \\ 218 & 255 \end{vmatrix}$	70	54	83	70
80	$\begin{vmatrix} 529 & 76 \\ 255 & 240 \end{vmatrix}$	70	48	87	76
85	$\begin{vmatrix} 529 & 99 \\ 231 & 241 \end{vmatrix}$	70	51	84	71
90	$\begin{vmatrix} 534 & 62 \\ 280 & 224 \end{vmatrix}$	72	49	90	78
95	$\begin{vmatrix} 569 & 40 \\ 304 & 187 \end{vmatrix}$	69	38	93	82

Tabulka 5.12– Výsledky neuronové sítě

Typ aktivační funkce:		ReLU			
Použitá data:		Set parametrů 2, kontinuální			
Threshold (%)	Confusion matice (-)	Přesnost (%)	Sensitivita (%)	Specificita (%)	Preciznost (%)
5	$\begin{vmatrix} 21 & 616 \\ 2 & 461 \end{vmatrix}$	44	100	3	43
10	$\begin{vmatrix} 58 & 567 \\ 17 & 458 \end{vmatrix}$	47	96	9	45
15	$\begin{vmatrix} 94 & 501 \\ 45 & 460 \end{vmatrix}$	50	91	16	48
20	$\begin{vmatrix} 149 & 467 \\ 63 & 421 \end{vmatrix}$	52	87	24	47
25	$\begin{vmatrix} 270 & 345 \\ 95 & 390 \end{vmatrix}$	60	80	44	53
30	$\begin{vmatrix} 301 & 292 \\ 126 & 381 \end{vmatrix}$	62	75	51	57
35	$\begin{vmatrix} 363 & 269 \\ 149 & 319 \end{vmatrix}$	62	68	57	54
40	$\begin{vmatrix} 395 & 223 \\ 166 & 316 \end{vmatrix}$	65	66	64	59
45	$\begin{vmatrix} 448 & 178 \\ 190 & 284 \end{vmatrix}$	67	60	72	61
50	$\begin{vmatrix} 501 & 140 \\ 210 & 249 \end{vmatrix}$	68	54	78	64
55	$\begin{vmatrix} 522 & 101 \\ 263 & 214 \end{vmatrix}$	67	45	84	68
60	$\begin{vmatrix} 551 & 94 \\ 269 & 186 \end{vmatrix}$	67	41	85	66
65	$\begin{vmatrix} 550 & 65 \\ 305 & 180 \end{vmatrix}$	66	37	89	73
70	$\begin{vmatrix} 578 & 50 \\ 310 & 162 \end{vmatrix}$	67	34	92	76
75	$\begin{vmatrix} 576 & 45 \\ 364 & 115 \end{vmatrix}$	63	24	93	72
80	$\begin{vmatrix} 574 & 40 \\ 364 & 122 \end{vmatrix}$	63	25	93	75
85	$\begin{vmatrix} 604 & 25 \\ 389 & 82 \end{vmatrix}$	62	17	96	77
90	$\begin{vmatrix} 615 & 20 \\ 405 & 60 \end{vmatrix}$	61	13	97	75
95	$\begin{vmatrix} 616 & 5 \\ 442 & 37 \end{vmatrix}$	59	8	99	88

Tabulka 5.13– Výsledky neuronové sítě

Typ aktivační funkce:		ReLU			
Použitá data:		Set parametrů 2, binární			
Threshold (%)	Confusion matice (-)	Přesnost (%)	Sensitivita (%)	Specificita (%)	Preciznost (%)
5	$\begin{vmatrix} 241 & 376 \\ 57 & 426 \end{vmatrix}$	61	88	39	53
10	$\begin{vmatrix} 315 & 325 \\ 83 & 377 \end{vmatrix}$	63	82	49	54
15	$\begin{vmatrix} 343 & 298 \\ 73 & 386 \end{vmatrix}$	66	84	54	56
20	$\begin{vmatrix} 324 & 285 \\ 111 & 380 \end{vmatrix}$	64	77	53	57
25	$\begin{vmatrix} 386 & 253 \\ 110 & 351 \end{vmatrix}$	67	76	60	58
30	$\begin{vmatrix} 409 & 227 \\ 120 & 344 \end{vmatrix}$	68	74	64	60
35	$\begin{vmatrix} 398 & 204 \\ 161 & 337 \end{vmatrix}$	67	68	66	62
40	$\begin{vmatrix} 386 & 209 \\ 160 & 345 \end{vmatrix}$	66	68	65	62
45	$\begin{vmatrix} 457 & 184 \\ 160 & 299 \end{vmatrix}$	69	65	71	62
50	$\begin{vmatrix} 450 & 170 \\ 160 & 320 \end{vmatrix}$	70	67	73	65
55	$\begin{vmatrix} 471 & 158 \\ 203 & 268 \end{vmatrix}$	67	57	75	63
60	$\begin{vmatrix} 499 & 131 \\ 188 & 282 \end{vmatrix}$	71	60	79	68
65	$\begin{vmatrix} 494 & 121 \\ 241 & 244 \end{vmatrix}$	67	50	80	67
70	$\begin{vmatrix} 509 & 114 \\ 220 & 257 \end{vmatrix}$	70	54	82	69
75	$\begin{vmatrix} 512 & 106 \\ 242 & 240 \end{vmatrix}$	63	50	71	54
80	$\begin{vmatrix} 518 & 76 \\ 286 & 220 \end{vmatrix}$	67	43	87	74
85	$\begin{vmatrix} 541 & 71 \\ 317 & 171 \end{vmatrix}$	65	35	88	71
90	$\begin{vmatrix} 590 & 41 \\ 357 & 112 \end{vmatrix}$	64	24	94	73
95	$\begin{vmatrix} 612 & 19 \\ 382 & 87 \end{vmatrix}$	64	19	97	82

Z tabulky 5.14 je patrné, že hodnoty součinu sensitivity a specificity nabývají mnohem lepších hodnot v případě neuronových sítí. Také je vidět, že hodnoty součinů jsou vždy větší při aplikaci parametrických setů v binárním formátu. Nevětší hodnotu součinu vykazuje neuronová síť s aktivační funkcí ReLU, která byla aplikována na parametrický set 1 v binárním formátu.

Tabulka 5.14: Maximální hodnoty součinu sensitivity a specificity pro jednotlivé ML modely

Typ ML modelu	Použitá data	Threshold (%)	Součin (-)
Logistická regrese	Set 1, kontinuální	55	0,13
Logistická regrese	Set 2, kontinuální	65	0,15
Neuronová síť – Sigmoid	Set 1, kontinuální	40	0,43
Neuronová síť – Sigmoid	Set 1, binární	35	0,51
Neuronová síť – Sigmoid	Set 2, kontinuální	45	0,46
Neuronová síť – Sigmoid	Set 2, binární	45	0,50
Neuronová síť – ReLU	Set 1, kontinuální	35	0,45
Neuronová síť – ReLU	Set 1, binární	65	0,52
Neuronová síť – ReLU	Set 2, kontinuální	45	0,43
Neuronová síť – ReLU	Set 2, binární	30	0,48

6 Diskuse

V rámci této bakalářské práce byly navrženy celkem tři ML metody pro vyhodnocení parametrických hodnot extrahovaných z EKG záznamů pacientů s potentním rizikem výskytu VF. Výběr těchto metod byl proveden na základě rešerše a následně byly poté vybrány ty ML metody, které se dle konzultované literatury nejvíce hodily pro daný klasifikační problém. Metody byly implementovány v jazyce Python ve vývojovém prostředí Spyder 3 a následně aplikovány na data získána z měření EKG 408 pacientů.

6.1 Metody předzpracování dat

Dataset, se kterým bylo v průběhu práce manipulováno, obsahoval celkem EKG data od 583 pacientů. Měřil se vždy klasický EKG signál, záznam byl pořízen vždy neprodleně po zjištění průběhu infarktu myokardu a byl ukončen buď po úspěšném aplikování léčby či po prodělání VF. Délky záznamů pro jednotlivé pacienty se tak od sebe lišily. Z těchto signálů bylo potom následně extrahováno celkem 20 číselných parametrů, jejichž úkolem bylo co nejvíce charakterizovat proces depolarizace a repolarizace komor srdce v rámci původního EKG signálu.

Tato parametrická data v některých případech v sobě částečně zahrnovala informaci o časové variabilitě původní EKG křivky, většina této informace se však extrahováním těchto parametrů zcela ztratila, jelikož výsledné parametrické hodnoty charakterizují původní EKG signál pouze pomocí průměrných hodnot. Prozatím všechny ML modely (uvedené v seznamu použité literatury) navržené pro analýzu a klasifikaci EKG v kontextu různých tachykardií pracovaly s kompletními EKG záznamy, a nikoliv pouze s parametry. Je dosti pravděpodobné, že právě informace o časové variabilitě obsažené např. v parametru jako HRV (Heart Rate Variability) či jiných frekvenčně závislých parametrech mohou být při predikci srdečních tachykardií zásadní či ne-li dokonce elementární.

Proces předzpracování dat komplikovala skutečnost, že chybějící data byla v původním datasetu značena odlišně. V některých případech hodnota chyběla zcela, v některých byla nahrazena hodnotou „NaN“ nebo zcela neadekvátní číselnou hodnotou. Tuto hodnotu se dalo považovat za neadekvátní, jelikož se těchto hodnot v daném záznamu pro konkrétní parametr nacházelo hned několik a jejich hodnota převyšovala ostatní normální naměřené hodnoty o několik jednotek řádů.

Pro účely aplikace ML modelů na dataset bylo potřeba zbavit se chybějících hodnot. V případě chybějících hodnot by ML model nebyl schopen správně interpretovat tuto hodnotu a program by hlásil chybu. Naopak velmi vysoké hodnoty oproti zbylým by velmi narušily vnitřní proces optimalizace ML modelu a ten by tak poté vykazoval chybné

výsledky. Z důvodu velké nekonzistence značení chybějících hodnoty bylo třeba tedy veškeré neadekvátní číselné či NaN hodnoty manuálně v programu Microsoft Excel přenastavit na prázdné hodnoty. V samotném Python kódu se poté nachází samostatný blok, který zaručuje filtraci tohoto již upraveného datasetu, kdy veškeré prázdné hodnoty jsou interpretovány jako NaN hodnoty. Následně jsou poté automaticky vyřazeni z datasetu ti pacienti, u kterých se alespoň jedna hodnota parametru nevyskytuje, resp. nabývá hodnoty NaN. Takto byl původní dataset redukován o záznamy od 175 pacientů, tedy z původních 583 záznamů zbylo pouze 408, jak je uvedeno v tabulce 5.1. Z ML hlediska by tato celková ztráta neměla představovat žádný zásadní problém, jelikož bylo úspěšně navrženo několik ML klasifikátorů či neuronových sítí, které pracovali s daty od několikanásobně méně pacientů než v tomto případě.

6.2 Návrh metod pro statistické zhodnocení

Úspěšná implementace ML klasifikačního modelu předpokládá, že se data patřící do jednotlivých kategorií budou nějakým statisticky významným způsobem lišit. Data byla nejdříve podrobena Kolmogotov – Smirnově testu pro ověření normality, jelikož původně bylo zamýšleno použít více průkazný statistický test než Mann – Whitney U test (např. nepárový T test). Většina těchto statisticky silnějších testů však předpokládá normální rozložení testovaných dat. Rozložení každého parametru bylo testováno zvláště, jelikož nebyl žádný důvod předpokládat, že data mezi jednotlivými parametry spolu nějak souvisejí, a tudíž data pro každý parametr mohla nabývat jiného rozložení.

Z výsledků Kolmogotov – Smirnova testu vyplynulo, že veškeré parametry nabývaly jiného než normálního rozdělení. Z důvodu nesplnění podmínky normality byl použit právě neparametrický Mann – Whitney U test pro důkaz odlišného rozdělení, přičemž byla použita standardní hodnota hladiny významnosti $\alpha = 5\ \%$. V tabulce 5.2 jsou poté uvedeny výsledky tohoto testu. Pomocí Mann – Whitney U testu se mezi sebou vždy porovnávalo rozložení dat náležící fibrilujícím pacientům s rozložením dat pro nefibrilující pacienty. U celkem pěti parametrů (HR, Tarea_V1, Tarea_V3, Tarea_V3, Tarea_I – vysvětlivky těchto značek jsou uvedeny v tabulce 4.1) rozdílná distribuce nebyla prokázána. Zbylých 15 parametrů prošlo testem a nadále bylo pracováno pouze s daty náležící těmto parametrům. Výsledek Mann – Whitney U testu také reprezentují grafy rozložení těchto zbylých 15 parametrů (v příloze B), kdy při okometrickém zhodnocení lze usoudit, že jednotlivé distribuce se od sebe lehce odsunuty.

Jelikož pro několik parametrů bylo potvrzeno rozdílné rozlišení s větší statistickou průkazností (*p* hodnota při testování těchto parametrů nabývala hodnot rovných či menších než 0,001), rozhodl jsem se vytvořit dva sety těchto parametrů. První tento set (označen jako set parametrů 1) obsahoval všechny parametry, u kterých bylo prokázáno rozdílné rozdělení nehledě na hodnotu *p.* Druhý set (označen jako set parametrů 2) poté obsahoval pouze výše zmíněné parametry s velmi nízkou hodnotou $p\ (p \leq 0{,}001)$. Přehled

jednotlivých parametrů obsažených v těchto setech je uveden v tabulce 5.3. Cílem vytvoření tohoto druhého setu parametrů bylo zlepšení klasifikačních schopností dále implementovaných ML metod. Je důvod předpokládat, že parametry s nízkou hodnotou *p* jsou více vhodné pro klasifikační účely. Jelikož jsou ML modely citlivé na veškeré hodnoty, kterým jsou vystaveny, a jejich výkonnost je odrazem kvality těchto vstupních dat, mohli v tomto případě parametry s vyšší hodnotou *p* „škodit" vstupním datům a jejich kvalita tak mohla být oproti druhému setu dat snížena. Považuji za vhodné uvést, že výkonnost ML modelů byla testována pro jednotlivé parametry separátně, takto dosažené výsledky však nabývali mnohem horších hodnot než v případě kombinace většího množství parametrů, resp. ML model takto úplně ztrácel schopnost predikovat.

Pro dosažení co nejlepších výsledků bylo nutno experimentovat s formátem vstupních dat. Původní dataset sestával kontinuálních hodnot jednotlivých parametrů. Byla tedy provedena ROC analýza parametrů, u kterých bylo již prokázáno rozdílné rozložení. Výstupem tohoto testu byly jednotlivé ROC křivky, každá pro jeden parametr. Na základě těchto křivek (v příloze A) byla volena cutoff (mezní) hodnota pro jednotlivé parametry, kdy hodnotám vyšší, než cutoff hodnota byla přiřazena hodnota 1 a hodnotám menším hodnota 0. Takto byl vytvořen druhý soubor s binárním typem dat. S ohledem na charakter problematicky klasifikace VF by bylo nejlepší, kdyby se zvolily takové hodnoty, které zajišťují co možná největší míru sensitivity z důvodu minimalizace případné nesprávné klasifikace fibrilujícího pacienta (FN). Na základě ROC analýzy byla definována optimální cutoff hodnota s maximálně možnými veličinami senzitivity a specificity. Jelikož velkých hodnot sensitivity dosahovaly ROC křivky až při velmi malých hodnotách specificity, bylo potřeba volit hodnotu cutoff na základě kompromistu mezi hodnotami senzitivity a specificity. V případě zvolení cutoff hodnot pro malé hodnoty specificity by ML model nebyl schopen skoro vůbec klasifikovat nefibrilující pacienty a drtivou většinu by jich zařadil nesprávně do fibrilující skupiny (FP).

6.3 Výsledky navržených ML modelů

Před použitím ML modelů pro klasifikaci bylo potřeba vyřešit jeden zásadní problém, který považuji za hlavní nedostatek datasetu. Jak je z tabulky 5.1 patrné, rozdělení počtu fibrilujících a nefibrilujících pacientů bylo v extrémně nerovnoměrné. Poměr počtu dvou tříd sice odpovídá skutečnosti, kdy minoritní počet pacientů nakonec prodělá VF, ale pro účely ML je tento disbalanc zásadním problémem. Tento nepoměr zapříčinil to, že ML model byl z drtivé většiny vystaven datům z nefibrilující skupiny, což nakonec vyústilo v potlačení vlivu hodnot fibrilujících pacientů. Tuto teorii potvrzují i předem provedené výpočty, kdy byly modely trénovány pomocí celého trénovacího setu a nebyly prakticky vůbec schopny klasifikovat fibrilující pacienty, tedy jejich sensitivity dosahovaly téměř nulových hodnot. Tento problém byl řešen pomocí vzorkování převažující nefibrilující skupiny. Vzhledem k počtu fibrilujících pacientů ve výsledném datasetu (23) byla zvolena hodnota počtu pacientů v jednom vzorku jako 30 proto, že byla srovnatelná

s hodnotou počtu fibrilujících pacientů, ale přitom stále lehce simulovala větší množství nefibrilujících pacientů v klinické praxi.

Při procesu tréninku ML modelu byl vstupní set dat rozdělen náhodně v poměru 8:2, kdy 80 % dat bylo použito pro natrénování ML modelu a zbylých 20 % k následnému testování. Těmto 20 % testovacích dat tedy ML model nebyl v procesu tréninku vůbec vystaven, čímž se zajistila reprezentativnost následných výsledků. Vstupní set však obsahoval data pouze od 53 pacientů (30 + 23), a při náhodném rozdělení dat na trénovací a testovací sadu se při jednotlivých procesech složení těchto sad zásadně lišilo, protože na testovací set připadalo vždy pouze 10 až 11 hodnot. Poměr počtů fibrilujících a nefibrilujících pacientů v rámci tohoto setu se zásadně lišil při každém novém náhodném rozdělení, a tím pádem se velmi lišily jednotlivé hodnoty metrických parametrů. K zajištění celkové statistické reprezentativnosti výsledků proběhlo při každém procesu klasifikace 100 iterací, aby se tyto hodnoty poté náležitě zprůměrovaly.

6.3.1 Návrh modelu logistické regrese

Binominální LR byla první a nejjednodušším navrženým modelem. Oproti později navrženým neuronovým sítím se vyznačuje jednoduchostí a mnohem větší výpočetní rychlostí. Z důvodu principu LR nebylo možno použít binárního formátu dat, jelikož tato metoda vyžaduje kontinuální rozložení jednotlivých vstupních hodnot a na tomto základě je poté navržena sigmoid křivka, která se snaží dané dvě třídy vstupních dat od sebe co nejlépe separovat, viz obrázek 4.4. Pokud bychom tedy přiřadili vstupním hodnotám pouze binární hodnoty, způsob modelace pomocí sigmoid křivky by ztrácel význam, jelikož by vždy zůstala stejně umístěna s ohledem k x – ové ose, kde by její střed ležel v bodě (0,5;0,5).

Z výsledků klasifikace pomocí ML modelu LR uvedených v tabulkách 5.4 a 5.5 je vidět, že na rozdíl od neuronových sítí stoupala kontinuálně přesnost modelu po celou dobu zvyšování hodnoty thresholdu. V extrémních hodnotách thresholdu však buď hodnota sensitivity či specificity klesla velmi nízko a model tak klasifikoval jednostranně. Vysoká hodnota přesnosti je zapříčiněna tím, že model se pro hodnotu thresholdu 95 % choval skoro jednostranně a klasifikovat téměř všechny pacienty jako nefibrilující. Pro střední hodnoty thresholdu byly obě hodnoty specificity a sensitivity poměrně malé což se také projevuje na zdaleka nejmenších hodnotách součinu sensitivity a specificity, viz. tabulka 5.14 Dle očekávání vykazoval ML model LR lepší výsledky při použití parametrického setu 2, hodnoty metrických parametrů mezi dvěma sety se však lišily jen lehce, o čemž svědčí i hodnoty součinů pro oba sety (0,13 pro set 1 a 0,15 pro set 2) Tato nízká výkonnost je z největší pravděpodobností zapříčiněna tím, že dvě třídy parametrických hodnot nejsou jednoduše lineárně separabilní, což je hlavním předpokladem pro úspěšné implementování LR.

6.3.2 Návrh modelů neuronových sítí

Dalšíma dvěma ML metodami, které byly navrhnuty, byly neuronové sítě. Tyto sítě sdíleli stejnou architekturu a lišili se pouze druhem aktivační funkce pro perceptrony ve skryté vrstvě. Aktivační funkce byla použita, jelikož se jeví jako standard při návrhu neuronových sítí pro klasifikaci binárního typu dat. Funkce ReLU je poté považována za více účinný typ aktivační funkce z důvodu absence horního limitu. Jelikož tato funkce nabývá lineárního tvaru pro veškeré hodnoty blížící se hodnotě $x = 0$ zprava, není omezena žádnou limitní hodnotou, ke které spěje, viz obrázek 4.12. Tato její charakteristická vlastnost ji zásadně odlišuje od aktivační funkce sigmoid, jelikož velké hodnoty nesaturují. V případě funkce sigmoid se interval největší citlivosti nachází okolo bodu $x = 0{,}5$, kde hodnoty derivace této funkce nabývají největších hodnot, viz obrázek 4.11. Parametrické hodnoty v datasetu se lišily o několik řádů, a i přes jejich normalizaci v průběhu algoritmu je pravděpodobné, že některé tyto parametrické hodnoty mohly nabývat v poměru k ostatním pořád extrémně nízkých či vysokých hodnot. V tomto případě by se poté nacházely v krajních oblastech na křivce funkce sigmoid, kde citlivost již není tak velká. Jinými slovy by tyto hodnoty saturovaly. Z tohoto důvodu se funkce ReLU jeví jako vhodnější volba, jelikož právě její charakteristický tvar zamezuje výše popsané situaci.

Při pohledu na výsledky klasifikace pomocí neuronových sítí (tabulky 5.6 až 5.13) je vidět zlepšení celkové schopnosti modelů predikovat VF oproti modelu LR. Neuronové sítě se oproti LR modelu vyznačují celkově vyšší mírou přesnosti, která je navíc relativně konzistentní pro všechny hodnoty thresholdů. Také hodnoty senzitivity a specificity jsou vyšší než u LR modelu navíc ve adekvátnějším poměru, o čemž také svědčí mnohem větší hodnoty součinů sensitivity a specificity, viz tabulka 5.14. Hlavním faktorem tohoto částečného úspěchu neuronových sítí je jednoznačně jejich schopnost zpětné vazby prostřednictvím postupné změny váhových koeficientů a hodnot biasů perceptornů v rámci neuronové sítě. Tato schopnost umožňuje neuronovým sítím se lépe přizpůsobit vstupním datům i v případě jejich komplexního typu rozložení. Z tohoto důvodu jsou při procesu optimalizace neuronové sítě mnohem flexibilnější nežli model logistické regrese, u kterého je proces optimalizace zajištěn pouze pomocí změny umístění senzitivního úseku křivky sigmoid funkce.

Výsledná architektura neuronových sítí je výsledkem heuristických pravidel, která byla zmíněna v použité literatuře a v průběhu testování výkonnosti neuronových sítí se jevila jako platící. Při navrhování architektury neuronové sítě je potřeba dle těchto pravidel určit počet skrytých vrstev a také počet perceptronů v těchto jednotlivých skrytých vrstvách. Jedno heuristické pravidlo říká, že počet perceptronů ve skrytých vrstvách by se měl pohybovat okolo hodnoty aritmetického průměru součtu vstupních a výstupních perceptronů. V průběhu testování se nejvíce osvědčil počet pěti perceptronů ve skryté vrstvě, což je hodnota pouze lehce menší nežli průměr. Dále pak bylo využito pouze jedné skryté vrstvy, jelikož další heuristické pravidlo tvrdí, že většina

klasifikačních binominálních problémů vyžaduje pro nejlepší výsledky pouze jednu skrytou vrstvu. Toto pravidlo bylo opět experimentálně potvrzeno, jelikož se zvyšujícím se počtem skrytých vrstev klesala schopnost predikce neuronové sítě.

Celkově byly neuronové sítě schopny vykazovat výsledky s jasně větší konzistencí, než tomu bylo v případě logistické regrese. O tom také svědčí hodnoty součinů specificit a senzitivit, viz tabulka 5.14. Oproti modelu LR se prokázal parametrický set 1 jako více vhodný. Lze tak polemizovat o možnosti, že kdyby bylo použito více parametrů, s jejich množstvím by se dále zvyšovala přesnost predikce. Nelepší celkové výsledky vykazovala neuronová síť s aktivační funkcí ReLU, která byla aplikována na parametrický set 1, a to v binární verzi (hodnota součinu 0,52). Téměř stejnou výkonnost také prokázala neuronová sít s aktivační funkcí sigmoid aplikována na stejný set parametrů (hodnota součinu 0,51). Z tohoto důvodu je vhodné se domnívat, že právě binární typ hodnot je možno použít s největší efektivitou. Na rozdíl od modelu LR se neuronové sítě vyznačovaly téměř konzistentní hodnotou přesnosti pro všechny hodnoty thresholdů, přičemž nejvíce přesné byly pro střední hodnoty thresholdů.

6.3.3 Závěr kapitoly

Závěrem lze polemizovat o možnosti využití klasifikace EKG záznamů na základě extrahovaných parametrů pro detekci fibrilujících pacientů. Zatímco model LR se jeví pro tuto úlohu jako nevhodný, z výsledků neuronových sítí je vidět jistý potenciál. Při použití neuronových sítí, kompletního datasetu v binárním formátu a hodnot thresholdů, které se ukázaly jako nejvíce vhodné, lze dosáhnout poměrně vysokých hodnot sensitivity a specificity, viz tabulka 6.1. Tyto hodnoty sice nejsou uspokojivě vysoké, aby mohly být využité jak komerčně, tak v klinické praxi, na druhou stranu již nechybí mnoho k uspokojivé hodnotě přesnosti a výsledky byla prokázána částečná vypovídající schopnost extrahovaných parametrů predikovat VF.

Tabulka 6.1: Přehled modelů neuronových sítí vykazující nejlepší výsledky

Typ aktivační funkce	Typ dat	Threshold (%)	Sensitivita (%)	Specificita (%)
Sigmoid	binární	65	79	65
ReLU	binární	35	65	81

Celkově je hlavní zjištění práce takové, že konkrétní formát parametrických dat, se kterýma bylo pracováno, není dostatečně vhodný pro zcela úspěšné aplikovaní ML metod. Přesnosti navržených neuronových sítí nelze srovnávat s jinými navrženými sítěmi, které pracovaly s kompletním záznamem EKG signálu, jelikož zde se hodnoty specificity a sensitivity pohybovaly obvykle okolo 95 %. Tato práce navrhuje buď použít parametrická data jako data pomocná, jelikož není vyloučeno, že by v kombinaci

s kompletním záznamem mohly ještě vylepšit výkon již stávajících neuronových sítí. Pro úspěšnější implementaci ML modelů pouze na základě parametrických hodnot tato práce navrhuje obstarat nová více průkazná data, u kterých bude možno lépe separovat jednotlivé hodnoty fibrilujících a nefibrilujících pacientů. Z důvodu velkého nepoměru mezi počtem fibrilujících a nefibrilujících pacientů bylo možno využít pro jeden tréninkový proces pouze malou část z celkového počtu dat. Pokud by byl ML model vystaven většímu počtu dat, kdy by jednotlivé třídy byly zastoupeny rovnoměrně, zlepšila by se tak pravděpodobně výkonnost jednotlivých ML modelů.

7 Závěr

Byla získána parametrická data od 583 pacientů s potenciálem prodělání VF. Tato data byla v průběhu procesu předzpracování zbavena neadekvátních a chybějících hodnot, čímž se původní dataset redukoval na záznam od 408 pacientů. Následně byla provedena statistická analýza těchto dat. Nejdříve byla data jednotlivých parametrů podrobena Kolmogotov – Smirnově testu normality. Z tohoto testu vyplynulo, že data žádného z těchto parametrů nemají normální rozložení, a tak byl zvolen Mann – Whitney U test pro ověření rozdílné distribuce dat fibrilující a nefibrilujícící skupiny pacientů.

U 15 z počátečních 20 byla pomocí tohoto testu prokázána rozdílná distribuce mezi skupinami. Na základě síly průkaznosti Mann – Whitney U testu byly vytvořeny dva sety těchto parametrů, přičemž první obsahoval všech 15 a druhý set obsahoval pouze ty parametry, u kterých byla průkaznost testu nejsilnější (*p* hodnota byla rovna či menší než 0,001). Výsledky klasifikace však ukazují, že u neuronových sítí byl s větší mírou přesnosti aplikován první set se všemi hodnotami, z čehož vyplývá, že kvantita vstupních parametrů je v případě použitého datasetu více důležitá než jejich kvalita v kontextu míry odlišnosti rozdělení fibrilující a nefibrilující třídy. Dále se také projevil binární formát vstupních dat jako více vhodný, kdy při jeho aplikaci ML modely dosahovaly lepších výsledků.

Byly navrženy celkem tři ML metody pro klasifikaci EKG parametrických záznamů. Těmito metodami byly: metoda LR a dvě neuronové sítě lišící se druhem aktivační funkce (sigmoid, ReLU). Z výsledků plyne, že model lineární regrese nelze úspěšně aplikovat na poskytnutá data, zatímco neuronové sítě jako klasifikátory mají potenciál pro úspěšnou aplikaci na EKG data. Hlavním výsledkem této práce je pak částečné potvrzení schopnosti extrahovaných parametrů sloužit jako indikátory VF při použití ML metod, avšak pro zcela průkazné výsledky doporučeno použít daná parametrická data jako data komplementární ke kompletnímu záznamu EKG signálu, na jehož základě bylo navrženo již několik velmi přesných ML metod. Navržené neuronové sítě společně s parametrickými daty je doporučeno použít jak podpůrné metody jiných přesnějších metod.

Dalším doporučením této práce je obstarat nový dataset parametrických hodnot, který by se od stávajícího lišil dvěma vlastnostmi. Zaprvé by měl obsahovat vyvážený počet záznamů pro fibrilující a nefibrilující pacienty, aby ho bylo možno v procesu tréninku ML modelu využít v celém rozsahu. Dále by pak mělo být zajištěno, aby jednotlivé parametrické hodnoty vykazovali průkazněji rozdílnou distribuci mezi pacienty, kteří prodělají a neprodělají VF. Je potřeba najít parametry nové, u kterých toto bude platit či získat data pro stávající parametry taková, kde budou více průkazná.

Seznam použité literatury

[1] HOLTE, Robert C. Ascent of machine learning in medicine. *Nature Materials* [online]. 2019, **18**(5), 407-407 [cit. 2020-01-07]. DOI: 10.1038/s41563-019-0360-1. ISSN 1476-1122. Dostupné z: http://www.nature.com/articles/s41563-019-0360-1

[2] SRINIVASAN, Neil T a Richard J SCHILLING. *Sudden Cardiac Death and Arrhythmias*. 2018, **7**(2). DOI: 10.15420/aer.2018:15:2. ISSN 2050-3369. Dostupné také z: https://www.aerjournal.com/articles/sudden-cardiac-death-and-arrhythmias

[3] LEE, Hyojeong, Soo-Yong SHIN, Myeongsook SEO, Gi-Byoung NAM a Segyeong JOO. Prediction of Ventricular Tachycardia One Hour before Occurrence Using Artificial Neural Networks. *Scientific Reports* [online]. 2016, **6**(1) [cit. 2019-10-24]. DOI: 10.1038/srep32390. ISSN 2045-2322. Dostupné z: http://www.nature.com/articles/srep32390

[4] TAYE, Getu Tadele, Eun Bo SHIM, Han-Jeong HWANG a Ki Moo LIM. Machine Learning Approach to Predict Ventricular Fibrillation Based on QRS Complex Shape. *Frontiers in Physiology* [online]. 2019, **10** [cit. 2019-10-24]. DOI: 10.3389/fphys.2019.01193. ISSN 1664-042X. Dostupné z: https://www.frontiersin.org/article/10.3389/fphys.2019.01193/full

[5] ACHARYA, U. Rajendra, Hamido FUJITA, Oh Shu LIH, Yuki HAGIWARA, Jen Hong TAN a Muhammad ADAM. Automated detection of arrhythmias using different intervals of tachycardia ECG segments with convolutional neural network. *Information Sciences* [online]. 2017, **405**, 81-90 [cit. 2019-10-24]. DOI: 10.1016/j.ins.2017.04.012. ISSN 00200255. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S0020025517306539

[6] VALUPADASU, Rama a Butchi Rama Rao CHUNDURI. Automatic Classification of Cardiac Disorders Using MLP Algorithm. *2019 Prognostics and System Health Management Conference (PHM-Paris)* [online]. IEEE, 2019, 2019, , 253-257 [cit. 2019-10-24]. DOI: 10.1109/PHM-Paris.2019.00050. ISBN 978-1-7281-0329-7. Dostupné z: https://ieeexplore.ieee.org/document/8756363/

[7] DEMIDOVA, Marina M., Alba MARTÍN-YEBRA, Jesper VAN DER PALS, Sasha KOUL, David ERLINGE, Pablo LAGUNA, Juan Pablo MARTÍNEZ a Pyotr G. PLATONOV. Transient and rapid QRS-widening associated with a J-wave pattern predicts impending ventricular fibrillation in experimental myocardial infarction. *Heart Rhythm* [online]. 2014, **11**(7), 1195-1201 [cit. 2020-04-19]. DOI: 10.1016/j.hrthm.2014.03.048. ISSN 15475271. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S1547527114003841

[8] HU, Wei, Xian JIN, Peng ZHANG, et al. Deceleration and acceleration capacities of heart rate associated with heart failure with high discriminating performance. *Scientific Reports* [online]. 2016, **6**(1) [cit. 2020-04-19]. DOI: 10.1038/srep23617. ISSN 2045-2322. Dostupné z: http://www.nature.com/articles/srep23617

[9] JOO, Segyeong, Kee-Joon CHOI a Soo-Jin HUH. Prediction of spontaneous ventricular tachyarrhythmia by an artificial neural network using parameters gleaned from short-term heart rate variability. *Expert Systems with Applications* [online]. 2012, **39**(3), 3862-3866 [cit. 2020-04-19]. DOI: 10.1016/j.eswa.2011.09.097. ISSN 09574174. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S0957417411014205

[10] LANE, R. E. Prediction and prevention of sudden cardiac death in heart failure. *Heart* [online]. 2005, **91**(5), 674-680 [cit. 2020-04-19]. DOI: 10.1136/hrt.2003.025254. ISSN 1355-6037. Dostupné z: http://heart.bmj.com/cgi/doi/10.1136/hrt.2003.025254

[11] PICCINI, Jonathan P., Min ZHANG, Karen PIEPER, et al. Predictors of sudden cardiac death change with time after myocardial infarction: results from the VALIANT trial. *European Heart Journal* [online]. 2010, **31**(2), 211-221 [cit. 2020-04-19]. DOI: 10.1093/eurheartj/ehp425. ISSN 0195-668X. Dostupné z: https://academic.oup.com/eurheartj/article-lookup/doi/10.1093/eurheartj/ehp425

[12] TERESHCHENKO, Larisa G., Barry J. FETICS, Peter P. DOMITROVICH, Bruce D. LINDSAY a Ronald D. BERGER. Prediction of Ventricular Tachyarrhythmias by Intracardiac Repolarization Variability Analysis. *Circulation: Arrhythmia and Electrophysiology* [online]. 2009, **2**(3), 276-284 [cit. 2020-04-19]. DOI: 10.1161/CIRCEP.108.829440. ISSN 1941-3149. Dostupné z: https://www.ahajournals.org/doi/10.1161/CIRCEP.108.829440

[13] DEMIDOVA, M.M., J. CARLSON, D. ERLINGE, J.E. AZAROV a P.G. PLATONOV. Prolonged Tpeak-Tend interval is associated with ventricular fibrillation during reperfusion in ST-elevation myocardial infarction. *International Journal of Cardiology* [online]. 2019, **280**, 80-83 [cit. 2020-04-19]. DOI: 10.1016/j.ijcard.2019.01.008. ISSN 01675273. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S016752731832518X

[14] DEMIDOVA, Marina M., Jonas CARLSON, David ERLINGE a Pyotr G. PLATONOV. Predictors of Ventricular Fibrillation at Reperfusion in Patients With Acute ST-Elevation Myocardial Infarction Treated by Primary Percutaneous Coronary Intervention. *The American Journal of Cardiology* [online]. 2015, **115**(4), 417-422 [cit. 2020-04-19]. DOI: 10.1016/j.amjcard.2014.11.025. ISSN 00029149. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S0002914914021663

[15] CHOLLET, François. *Deep learning with Python*. Shelter Island, NY: Manning, [2018]. ISBN 978-161-7294-433.

[16] 6 areas where artificial neural networks outperform humans. *Venturebeat* [online]. Ukit AI: Venturebeat, 2017 [cit. 2020-05-19]. Dostupné z: https://venturebeat.com/2017/12/08/6-areas-where-artificial-neural-networks-outperform-humans/

[17] BISHOP, Christopher M. *Pattern recognition and machine learning*. 2007. New York: Springer, c2006. Information science and statistics. ISBN 03-873-1073-8.

[18] SIMEONE, Osvaldo. A Very Brief Introduction to Machine Learning With Applications to Communication Systems. *IEEE Transactions on Cognitive Communications and Networking* [online]. 2018, **4**(4), 648-664 [cit. 2020-05-19]. DOI: 10.1109/TCCN.2018.2881442. ISSN 2332-7731. Dostupné z: https://ieeexplore.ieee.org/document/8542764/

[19] ALFARAS, Miquel, Miguel C. SORIANO a Silvia ORTÍN. A Fast Machine Learning Model for ECG-Based Heartbeat Classification and Arrhythmia Detection. *Frontiers in Physics* [online]. 2019, **7** [cit. 2020-01-07]. DOI: 10.3389/fphy.2019.00103. ISSN 2296-424X. Dostupné z: https://www.frontiersin.org/article/10.3389/fphy.2019.00103/full

[20] JOSEPHSON, Mark E. Sudden cardiac arrest. *Indian Heart Journal* [online]. 2014, **66**, S2-S3 [cit. 2019-10-24]. DOI: 10.1016/j.ihj.2014.01.001. ISSN 00194832. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S0019483214000078

[21] THIEBAUT, Cecile. Gradient descent for linear regression using Golang. *Backlog* [online]. Backlog: Backlog, 2019 [cit. 2020-05-19]. Dostupné z: https://backlog.com/blog/gradient-descent-linear-regression-using-golang/

[22] Logistic Regression with Python. *Medium* [online]. Medium: Medium, 2019 [cit. 2020-05-19]. Dostupné z: https://medium.com/@ODSC/logistic-regression-with-python-ede39f8573c7

[23] SVAČINOVÁ CSÉFALVAIOVÁ, Kornélia a Jitka LANGHAMROVÁ. Risk Factors of Severe Cognitive Impairment in the Czech Republic. SKIADAS, Christos H. a Charilaos SKIADAS, ed. *Demography and Health Issues* [online]. 1. Cham: Springer International Publishing, 2018, 2018-05-17, s. 267-273 [cit. 2020-05-19]. The Springer Series on Demographic Methods and Population Analysis. DOI: 10.1007/978-3-319-76002-5_22. ISBN 978-3-319-76001-8. Dostupné z: http://link.springer.com/10.1007/978-3-319-76002-5_22

[24] KOTOLOVÁ, Veronika. *Metody hodnocení zatížení a určení bezpečné struktury týmu v protivzdušné obraně*. Kladno, 2019. Diplomová práce. České vysoké učení technické v Praze. Vedoucí práce Ing. Jan Hejda, Ph.D.

[25] SAPORITO, Gerry. What is a perceptron? *Towardsdatascience* [online]. Towardsdatasciece: Towardsdatasciece, 2019 [cit. 2020-05-19]. Dostupné z: https://towardsdatascience.com/what-is-a-perceptron-210a50190c3b

[26] NIELSEN, Michael A. Neural Networks and Deep Learning. *Neuralnetworksanddeeplearning* [online]. Neuralnetworksanddeeplearning: Determination Press, 2015 [cit. 2020-05-19]. Dostupné z: http://neuralnetworksanddeeplearning.com/chap1.html

[27] AGARAP, Abien F. Deep Learning using Rectified Linear Units (ReLU). *Research Gate* [online]. , 2 [cit. 2020-05-19]. Dostupné z: https://www.researchgate.net/publication/323956667_Deep_Learning_using_Rectified_Linear_Units_ReLU

[28] Machine Learning Note: ReLU function. In: *Clay-Technology world* [online]. Clay-Technology world: Clay-Technology world, 2020 [cit. 2020-05-19]. Dostupné z: https://clay-atlas.com/us/blog/2020/02/03/machine-learning-english-note-relu-function/

[29] HEATON, Jeff. Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning. *Genetic Programming and Evolvable Machines* [online]. 2018, **19**(1-2), 305-307 [cit. 2020-05-19]. DOI: 10.1007/s10710-017-9314-z. ISSN 1389-2576. Dostupné z: http://link.springer.com/10.1007/s10710-017-9314-z

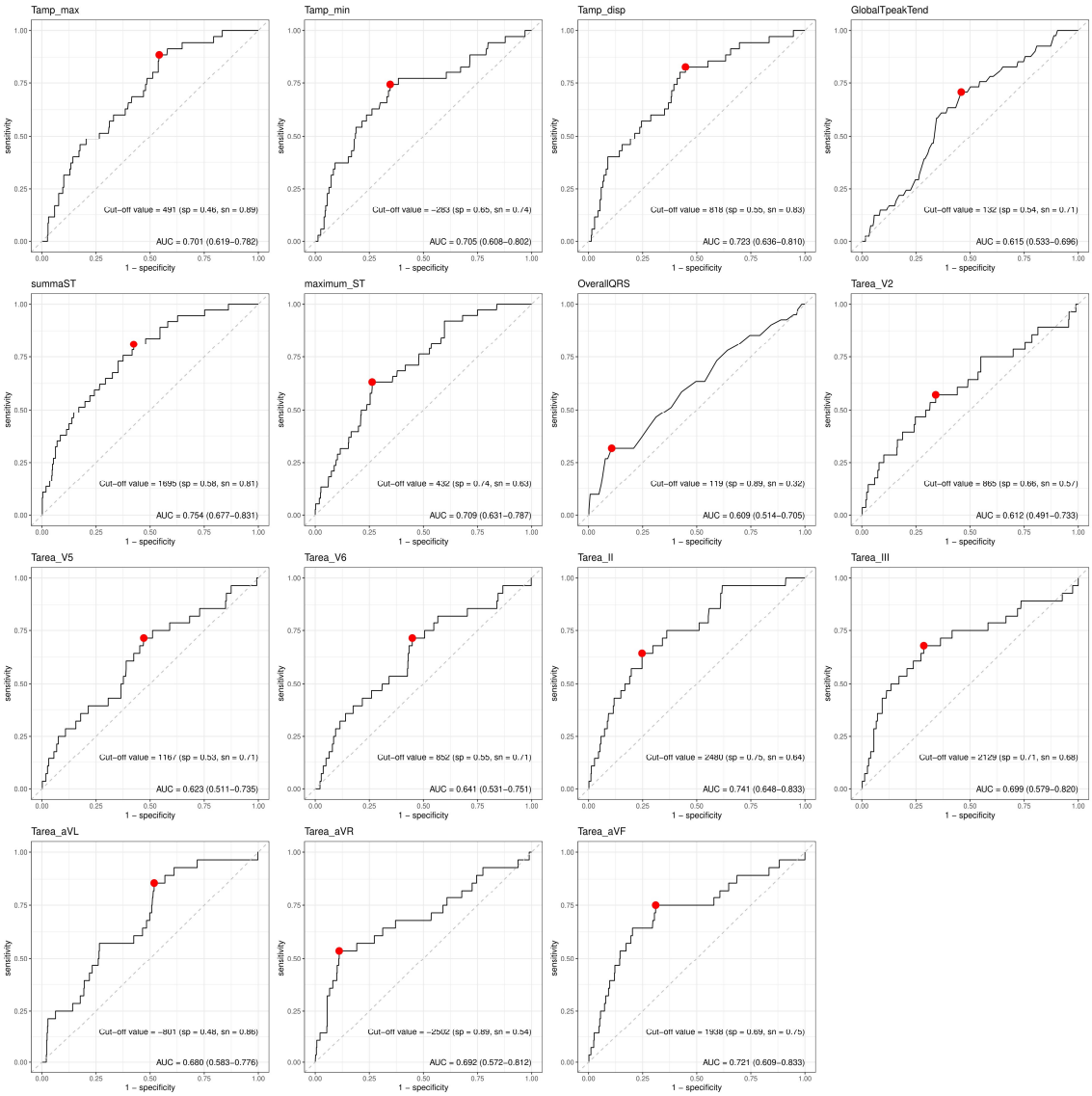
[30] Coding Deep Learning for Beginners — Linear Regression (Part 3): Training with Gradient Descent. *Towardsdatascience* [online]. Towardsdatasciece: Towardsdatasciece, 2018 [cit. 2020-05-19]. Dostupné z: https://towardsdatascience.com/coding-deep-learning-for-beginners-linear-regression-gradient-descent-fcd5e0fc077d

[31] What is gradient descent in machine learning? *Saugatbhattarai* [online]. saugatbhattarai: saugatbhattarai, 2020 [cit. 2020-05-19]. Dostupné z: https://saugatbhattarai.com.np/what-is-gradient-descent-in-machine-learning/

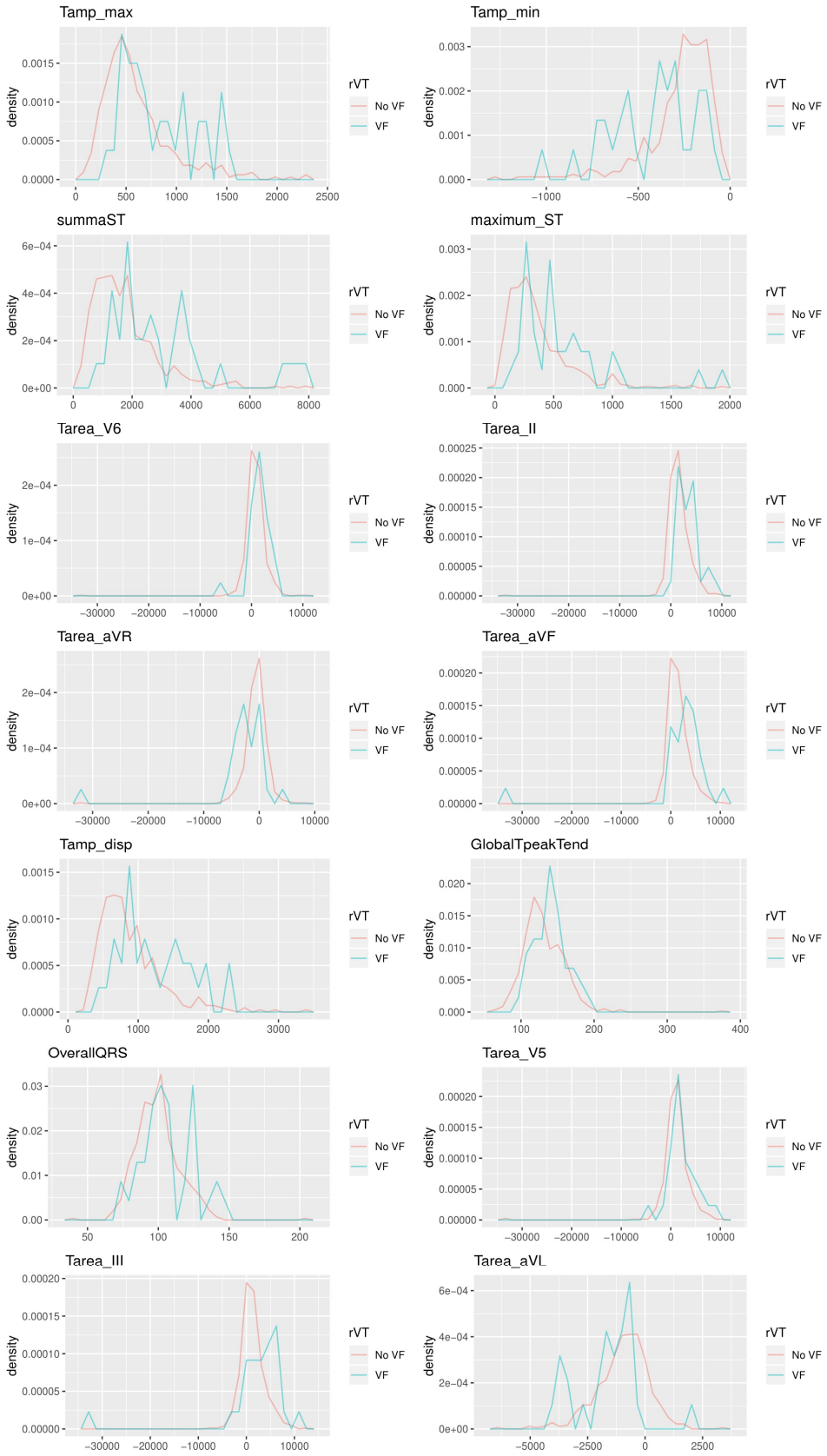
[32] Understanding Learning Rate. *Towardsdatascience* [online]. Towardsdatasciece: Towardsdatasciece, 2019 [cit. 2020-05-19]. Dostupné z: https://towardsdatascience.com/https-medium-com-dashingaditya-rakhecha-understanding-learning-rate-dd5da26bb6de

[33] JAPKOWICZ, Nathalie a Shaju STEPHEN. The class imbalance problem: A systematic study1. *Intelligent Data Analysis* [online]. 2002, **6**(5), 429-449 [cit. 2020-05-17]. DOI: 10.3233/IDA-2002-6504. ISSN 15714128. Dostupné z: https://www.medra.org/servlet/aliasResolver?alias=iospress&doi=10.3233/IDA-2002-6504

Příloha A: Grafy ROC křivek



Příloha B: Grafy rozložení parametrických dat



Příloha C: Obsah přiloženého DVD

1. Zadání bakalářské práce.pdf
2. Bakalářská práce.pdf
3. Abstrakt.pdf
4. Abstract.pdf
5. Klíčová slova, keywords.pdf
6. Složka souborů se zdrojovými kódy, jednotlivé kódy ve formátu .py