FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Rozpoznávání extrémistických textů
Student:	Pavlína Pokorová
Vedoucí:	Ing. Mgr. Ladislava Smítková Janků, Ph.D.
Studijní program:	Informatika
Obor / specializace:	Znalostní inženýrství
Katedra:	Katedra aplikované matematiky
Platnost zadání:	do konce letního semestru 2022/2023

Pokyny pro vypracování

Cílem bakalářské práce je navrhnout a implementovat vhodný klasifikátor pro detekci vybraných druhů extrémistických textů.

1. Seznamte se s metodami klasifikace textu a zpracujte rešerši.
2. Zpracujte data od externího konzultanta na problematiku extrémismu, event. jiné zdroje a vytvořte experimentální dataset obsahující ukázky neonacistických nebo džihádistických textů i neextrémistické texty.
3. Navrhněte/vyberte metodu, kterou použijete k řešení problému.
4. S využitím existujících nástrojů implementujte zvolenou metodu.
5. Navrhněte a proveďte experimenty, zpracujte jejich výsledky a proveďte jejich rozbor.

Elektronicky schválil/a Ing. Karel Klouda, Ph.D. dne 13. prosince 2021 v Praze.

Bakalářská práce

ROZPOZNÁVÁNÍ EXTREMISTICKÝCH TEXTŮ

Pavlína Pokorová

Fakulta informačních technologií
Katedra aplikované matematiky
Vedoucí: Ing. Mgr. Ladislava Smítková Janků, Ph.D.
23. června 2022

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů kódu

Tímto bych ráda poděkovala svému příteli a také své rodině za podporu nejen při psaní této práce. Poděkování patří také vedoucí práce, Ing. Mgr. Ladislavě Smítkové Janků, Ph.D., za cenné rady a pomoc při tvorbě datasetu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 citovaného zákona.

V Praze dne 23. června 2022 .....

Abstrakt

Tato práce je zaměřena na rozpoznávání extremistických textů s využitím modelů supervizovaného strojového učení. Cílem je vytvoření klasifikačního systému schopného detekovat vybrané druhy extremistických textů. V rámci této práce byly vytvořeny tři různé datasety českých textů, přičemž extremistický dataset obsahuje zejména neonacistické a nacistické texty. Navržený klasifikační systém vytváří predikce na základě váženého hlasování několika dílčích klasifikátorů. Tyto klasifikátory byly vytvořeny pomocí implementovaného trénovacího programu. Vhodné kombinace metod pro tvorbu klasifikátorů byly vybrány na základě provedených experimentů. Při těchto experimentech byla odhadována úspěšnost klasifikace textu v závislosti na použitých metodách pro předzpracování textu a extrakci příznaků. Pro klasifikaci textu byly využity dva různé klasifikační modely, SVM a Random forest. Přesnost výsledného klasifikačního systému je odhadována na 85 %.

Klíčová slova extremismus, klasifikace textu, strojové učení, zpracování přirozeného jazyka, extrakce příznaků, ladění hyperparametrů, random forest, support vector machine

Abstract

The focus of this thesis is recognition of extremist texts using supervised machine learning models. The goal is to create a classification system capable of specific extremist text detection. Three different datasets were created in this thesis, with the extremist one being focused mainly on nazi and neo-nazi texts. Proposed classification system creates predictions based on weighted votes of partial classificators. These classificators were created using an implemented training program. Appropriate combinations of methods for their creation were chosen based on performed experiments. In these experiments classification success rate was estimated based on used text preprocessing method and feature extraction method. Two classification models were used for text classification, specifically SVM and Random forest. Accuracy of the final classification system is estimated to be 85 %.

Keywords extremism, text classification, machine learning, natural language processing, feature extraction, hyperparameter tuning, random forest, support vector machine

Kapitola 1
Úvod

Šíření radikálních názorů je v online prostoru mnohem snazší díky anonymitě a velkému počtu uživatelů. Především sociální sítě jsou často využívány extremistickými skupinami k propagaci jejich ideologií a hledání nových příznivců. S nárůstem popularity sociálních sítí tak roste také počet extremistických skupin, které jejich prostřednictvím šíří svoji propagandu.

Aby bylo možné omezit počet extremistických příspěvků nejen na sociálních sítích, musíme být schopni takové příspěvky identifikovat. Vzhledem k jejich obrovskému množství je nutné využít metody automatické klasifikace textu. Sociální sítě již využívají různé techniky pro rozpoznávání teroristického či extremistického obsahu. Stále je zde však snaha o optimalizaci a zvýšení přesnosti metod, které se k tomuto účelu používají.

Cílem práce je vytvoření klasifikačního systému, který bude schopen detekovat extrémistické texty psané v českém jazyce. Tento systém je možné vytvořit mnoha různými způsoby, nicméně se v této práci omezíme pouze na metody klasifikace textu za pomoci modelů supervizovaného strojového učení.

Tento cíl můžeme rozdělit na několik dílčích úkolů. Prvním z nich je vypracování přehledu související literatury a rozbor používaných metod pro klasifikaci textu. Tímto úkolem se budeme zabývat v kapitole „Literární rešerše“. Následujícím úkolem je tvorba datasetu, kterou se budeme zabývat v praktické části práce. Poté na tomto datasetu provedeme několik experimentů, na základě kterých budou vybrány vhodné metody pro vytvoření klasifikátorů. Nakonec implementujeme samotný klasifikační systém a vyhodnotíme jeho přesnost.

Kapitola 2
Literární rešerše

2.1 Vymezení základních pojmů

Porozumění následujícím pojmům je důležité pro pochopení smyslu této práce. Pro žádný z nich v současné době neexistuje všeobecně uznávaná definice. Proto se v odborné literatuře setkáváme s různými výklady, které se od sebe mohou významně lišit.

2.1.1 Extremismus

Pro tento pojem se nabízí spousta různých definic. V poslední době jsou však zaměřeny spíše na násilný extremismus [1]. Ten můžeme chápat, jako ideologický směr, který podporuje využití násilí k dosažení daných cílů, které se obecně týkají sociální, či rasové problematiky, náboženství, nebo politiky. Násilný extremismus bývá často zaměňován s terorismem. Pojem terorismus však označuje samotný akt násilí, či vyhrožování jeho použitím, zatímco násilný extremismus je pouze ideologie, která takovéto jednání podporuje. [2]

Obecně však extremismus nemusí s násilím přímo souviset. Definovat tento pojem je tak mnohem náročnější [1]. Museli bychom zavést jasnou hranici oddělující přijatelné postoje a názory od těch extrémních. Ministerstvo vnitra České republiky definuje extremismus následovně: *„Pojmem extremismus jsou označovány vyhraněné ideologické postoje, které vybočují z ústavních, zákonných norem, vyznačují se prvky netolerance, a útočí proti základním demokratickým ústavním principům, jak jsou definovány v českém ústavním pořádku.“* [3]

2.1.2 Radikalizace

S extremismem souvisí také pojem radikalizace. Na radikalizaci se dá pohlížet, jako na proces změny přesvědčení a názorů člověka, který může vést k extremismu. Jedná se tedy o proces přejímání politických, sociálních, či náboženských ideálů, které jsou v rozporu se zavedenými normami. [2]

2.1.3 Propaganda

Propagandou rozumíme systematické šíření zkreslených, či přímo lživých informací, za účelem ovlivnění názorů čtenáře ve prospěch propagátora. Typicky obsahuje velmi jednostranný pohled na realitu a poskytuje pouze informace, které se autorovi hodí. Propaganda se vyznačuje také tím, že je cílena na širokou veřejnost. [4]

2.1.4 Nacismus

Nacionální socialismus neboli zkráceně nacismus je krajně pravicová ideologie, která byla uplatňována v Německu mezi lety 1933 a 1945 politickou stranou NSDAP. Nacistická ideologie je založena na odmítání demokratických, liberálních a humanistických hodnot. Jedná se o německou obdobu fašismu, která se odlišuje zejména silnými prvky rasismu a antisemitismu. Důležitým aspektem této ideologie je přesvědčení o nadřazenosti německého národa a árijské rasy. [5]

Adolf Hitler ve své knize Mein Kampf představil rozdělení společnosti podle ras do tří různých kategorií: zakladatele kultury, nositele kultury a ničitele kultury, přičemž německý národ řadil do první skupiny. Židé byli označováni, jako protiklad árijské rasy a patřili tedy do skupiny ničitelů kultury. Propaganda strany NSDAP prosazovala nadřazenost árijské rasy a její právo vládnout světu, přičemž národy, které byly považovány za podřadnou rasu odsuzovala k otroctví či vyhlazení. [5]

2.1.5 Neonacismus

Neonacismus ideologicky navazuje na rasovou koncepci Hitlerova Německa. Vyznačuje se zejména prvky militantního rasismu a antisemitismu. Dále je pro tuto ideologii charakteristický odpor k imigrantům, homosexuálům či zastáncům komunismu. Radikální neonacisté nesouhlasí s moderní společností a východisko vidí v rozpoutání Svaté Rasové Války, díky které by měl být zaveden nový společenský řád na rasovém, panárijském základě (panárijský znamená upřednostňující árijskou rasu). [6]

2.2 Vývoj metod detekce extremismu v online prostoru

V této sekci se seznámíme s vývojem používaných metod pro odhalování extremistických skupin a rozpoznávání extremistického obsahu na internetu. Sestavení tohoto přehledu předcházel průzkum mnoha publikací z předešlých let. Některé z nich si zde představíme a ukážeme na nich, jaké metody se pro odhalování extremismu v dané době používaly.

2.2.1 Zdroje dat

V roce 2006 byly nejčastějšími zdroji dat pro výzkum extremismu na internetu blogy, diskusní fóra a webové stránky extremistických skupin. Nárůst popularity Youtube a sociálních sítí však znamenal postupný přesun extremistických skupin právě na tyto platformy, které se následně staly zdrojem dat k identifikaci nenávistného obsahu a hledání vztahů mezi danými skupinami či jednotlivci. [7]

Zavedení metod automatické detekce extremistického obsahu na sociálních sítích přispělo k přesunu extremistických skupin na darknet. Mají zde možnost komunikovat a plánovat své akce včetně případných teroristických útoků v anonymitě. Sociální sítě se tak staly spíše prostředkem pro hledání nových členů, se kterými se dále komunikuje přes darknet. Proto zejména novější výzkumy často pracují s daty získanými z darknetu. [8]

2.2.2 Používané metody

První studie zaměřené na detekci extremismu na internetu používaly manuální metody identifikace extremistického obsahu. To znamená, že data byla postupně analyzována expertem, který rozhodoval, zda se jedná o extremistický obsah a případně do jaké spadá kategorie. Jako příklad zde můžeme uvést studii [9] z roku 2001, zaměřenou na výzkum metod používaných extremistickými skupinami pro nábor mladých lidí přes internet. Výsledků bylo v této studii dosaženo manuální analýzou a kategorizací dat, která byla pro tento účel sesbírána z webových stránek.

Manuální metody se v dnešní době samozřejmě nepoužívají pro klasifikaci velkých objemů dat. Jde o velmi zdlouhavý a neefektivní postup. [10]

Jednou z prvních publikací, které pro analýzu působení extremistických skupin na internetu využily automatizovaný přístup je tato [11] z roku 2005. Zde byla použita attribute-based analýza obsahu webových stránek. Tato metoda umožňuje přiřadit danému obsahu některý z předem definovaných atributů, jako například: propaganda, fundraising, nebo recruitment, podle definovaných pravidel. Dále zde byla využita analýza linků (link analysis) pro zjištění vztahů mezi webovými stránkami různých extremistických skupin.

Dále zde uvádíme výzkum [12] z roku 2009 zaměřený na identifikaci extremistických videí. Namísto analýzy samotných videí se zde pracovalo s textovými daty (názvy videí, jejich popisy a komentáři uživatelů). Při sběru dat byla vyhledávána videa podle klíčových slov souvisejících s různými extremistickým ideologiemi. Z takto nalezených videí byla extrahována textová data, která byla následné klasifikována s využitím modelu support vector machine (SVM). Jedná se o jedno z prvních využití metod strojového učení k identifikaci extremistického obsahu.

V roce 2014 vyšel článek [13], který zkoumal různé metody pro odhalování pokusů extremistických skupin o nábor nových členů na sociálních sítích. Pro detekci takovýchto příspěvků byly použity následující modely: naivní Bayesův klasifikátor, logistická regrese, rozhodovací stromy, boosting a SVM. Texty z použitého datasetu byly převedeny do vektorové podoby s využitím metody tf-idf. Úspěšnost klasifikace těchto modelů byla následně měřena a porovnávána. Nejlepších výsledků v tomto experimentu dosáhl model SVM.

Metody hlubokého učení (deep learning) se pro rozpoznávání extremistického obsahu začaly využívat kolem roku 2017. Jako příklad zde uvádíme publikaci [14] zaměřenou na identifikaci extremistické propagandy pomocí hlubokého učení. Zde je představena robustní metoda, která dokáže identifikovat extremistické texty v různých jazycích. Pro řešení problému byly navrženy tři plně propojené neuronové sítě s různou komplexitou.

V současnosti se k detekci extremismu využívají převážně metody strojového učení a neuronové sítě. Preferovány jsou spíš neuronové sítě, díky jejich schopnosti zapamatování kontextu a případných dalších informací, které mohou ovlivnit úspěšnost klasifikace. [10]

2.3 Automatická klasifikace textu

Se zvětšujícím se objemem textových dokumentů dostupných na internetu stoupá také potřeba jejich automatické analýzy a klasifikace. Klasifikace textu má široké uplatnění, například při analýze sentimentu, rozpoznávání emailových spamů či v oblasti information retrival. V současnosti se využívají zejména metody založené na strojovém učení a neuronových sítích. V této práci se omezíme na využití metod supervizovaného strojového učení pro detekci extremistických textů.

Strojové učení umožňuje vytvářet klasifikační či regresní modely, aniž bychom je museli pro daný problém explicitně programovat. Tyto modely se přizpůsobují datům ve fázi učení (trénování). Při nesupervizovaném učení neznáme hodnoty vysvětlované proměnné Y u trénovacích dat. Snažíme se hledat mezi těmito daty vzájemné vztahy a podobnosti, na základě kterých je můžeme členit do skupin. Naopak při supervizovaném strojovém učení jsou modely trénovány na datech s přidělenými hodnotami Y. Na základě těchto dat je vytvořen model, který zachycuje vztahy mezi hodnotami příznaků a vysvětlované proměnné. Takový model je pak schopen predikovat Y pro zadané příznaky. [15]

Klasifikace představuje problém členění dat do n kategorií, vysvětlovaná proměnná tedy nabývá pouze diskrétních n hodnot. Při regresi pracujeme s vysvětlovanou proměnnou, která nabývá tolika hodnot, že ji můžeme považovat za spojitou. V této části práce dále popíšeme metody a klasifikační modely používané při klasifikaci textu. [16]

2.3.1 Předzpracování textu

V angličtině tento krok označujeme, jako natural language processing (zkráceně NLP). Předzpracováním vstupního textu můžeme významně ovlivnit přesnost výsledné klasifikace. Cílem předzpracování je očištění vstupního textu od znaků a slov, které nemají pro klasifikaci význam a převedení slov do základního tvaru. Tím docílíme lepších výsledků při následné extrakci příznaků. Pro předzpracování se často používají níže uvedené techniky. [17]

2.3.1.1 Převedení textu na pouze malá písmena

Slova, která se vyskytují na začátcích vět mají rozdílný zápis od ostatních. Velikost písmen hraje roli při rozhodování, zda jsou daná slova považována za stejná. Vzhledem k tomu, že při extrakci příznaků často pracujeme s počty výskytů slov, je dobré tyto rozdíly odstranit. Častým řešením je převedení na pouze malá (případně velká) písmena. Tento přístup však může způsobit ztrátu některých informací. Například název fakulty „FIT“ je po převedení k nerozeznání od slova „fit“. [18]

2.3.1.2 Tokenizace

Tokenizace rozděluje text na menší části nazývané tokeny. Mohou jimi být jednotlivá slova či n-gramy, tedy n-tice po sobě jdoucích slov. Vstupní text je rozdělen na tokeny podle bílých znaků a interpunkce. Interpunkční znaménka mohou být tokenizací odstraněna, nebo oddělena od slov a považována za samostatné tokeny. Moderní tokenizery neoddělují interpunkci, která ovlivňuje význam slova, jako například apostrofy v anglickém jazyce, spojovníky a desetinné čárky. Tokeny obsahující pouze interpunkční znaménka je dobré následně odstranit, pokud k jejich odstranění nedošlo při tokenizaci automaticky. Takto získané tokeny jsou použity, jako vstupy pro další kroky předzpracování. [19, 20]

2.3.1.3 Odstranění stopslov

Stopslova jsou často se vyskytující slova, která nepřináší důležité informace pro klasifikaci, například: „a“, „v“, „se“, „že“. Tato slova mívají v textu pouze syntaktický význam. Můžeme je chápat jako šum, kvůli kterému je pro klasifikátor obtížené zaměřit se na relevantní slova. Jejich odstraněním snížíme počet příznaků a zároveň zvýšíme relevanci slov, která mají pro klasifikaci význam. [21]

Můžeme sestavit vlastní seznam stopslov podle počtu výskytů slov v trénovací množině, nebo použít některý z již existujících seznamů, které jsou dostupné pro různé jazyky včetně češtiny. [21]

2.3.1.4 Stemming

Různé tvary stejného slova jsou při extrakci příznaků považovány za rozdílné informace a pro každý takový tvar je tedy vytvořen samostatný příznak. Abychom mohli sjednotit různé tvary slov se stejným významem pod jeden příznak, musíme je nejprve převést do společného tvaru. Toho docílíme použitím stemmingu či lemmatizace. Cílem stemmingu je nalezení kmenu (stem) slova. Kmen slova je ta část, která se v různých tvarech slova nemění. Stemmery Používají definovaná pravidla k odstraňování předpon, přípon a koncovek jednotlivých slov. Kontext, ve kterém je slovo použito, ani jeho slovní druh zde nehrají roli. [22]

2.3.1.5 Lemmatizace

Cílem lemmatizace je nalezení normalizovaného tvaru slov. Na rozdíl od stemmingu, lemmatizace převádí slovo na takzvané „lemma“ až po zjištění jeho slovního druhu a kontextu, v jakém je ve

větě použito. Z kontextu je zjištěn význam slova, který určuje, na jaké lemma bude dané slovo převedeno. Lemma tedy reprezentuje základní tvar slova s daným významem.[22]

Lemmatizer je náročnější na implementaci než stemmer, jelikož zde zjišťujeme slovní druhy a významy slov z jejich kontextu. Ze stejného důvodu je také následný proces lemmatizace časově náročnější. Často však poskytuje lepší výsledky. Dokáže například rozlišit homonyma podle jejich významu, nebo naopak sloučit slova s podobným či stejným významem, která jsou jinak zapsaná. Správně navržený lemmatizer by měl například převézt slova „jít“ a „šel“ na stejný základní tvar. Stejně jako u stemmingu zde záleží na jazyce daného textu.[22, 23]

Tabulka 2.1 Porovnání výsledků stemmingu a lemmatizace

Původní slovo	Stemming	Lemmatizace
kontaktovat	kontakt	kontaktovat
kontakty	kontakt	kontakt
jsem	js	být
jsou	js	být
budeme	bud	být

2.3.2 Extrakce příznaků

Předzpracovaný text nyní převedeme do strojově zpracovatelné podoby. Existují různé metody, které umožňují převedení jednotlivých polí tokenů, na číselné vektory. Tuto vektorovou reprezentaci použijeme k trénování klasifikátoru a následné klasifikaci textu. Níže jsou popsány některé často používané modely pro extrakci příznaků.

2.3.2.1 Bag-of-words

Zkráceně BoW je nejintuitivnějším modelem pro extrakci příznaků. Reprezentuje text (neboli dokument) pomocí vektoru, který je tvořen počty výskytů jednotlivých slov či jiných tokenů. Vektory dokumentů pak tvoří matici, která obsahuje v řádcích vektorové reprezentace dokumentů a její sloupce reprezentují jednotlivá slova a jejich četnosti v daných dokumentech. Čím častěji se vyskytuje dané slovo v textu, tím má vyšší váhu. Proto je velmi důležité odstranění stopslov před použitím tohoto modelu. Pořadí slov v textu zde nehraje roli, a tak dochází ke ztrátě informací o vzájemném vztahu slov. Přesto se dá tímto způsobem docílit dobrých výsledků při klasifikaci.[17]

2.3.2.2 Term Frequency-Inverse Document Frequency

Pro tuto metodu používáme zkratku tf-idf. Používá se ke snížení váhy slov, které se v dokumentech často vyskytují. Častá slova totiž nebývají při klasifikaci příliš důležitá vzhledem k tomu, že se vyskytují ve velké části dokumentů. Jak napovídá název této metody, váhy jednotlivých slov jsou vypočteny pomocí kombinace dvou veličin:

- **Term frequency (tf)** je míra četnosti výskytů slova v dokumentu. Může být reprezentována přímo počtem výskytů slova či binárně tak, že 1 znamená alespoň jeden výskyt v dokumentu. Dají se zvolit i jiné způsoby, při volbě metody výpočtu záleží na řešeném problému.
- **Inverse document frequency (idf)** je míra toho, jak obvyklé je slovo napříč všemi dokumenty. Také zde existují různé varianty výpočtu. Můžeme použít například vzorec:

$$\mathrm{idf}(w, D) = \log\left(\frac{|D|}{|d \in D : w \in d|}\right), \tag{2.1}$$

kde w označuje dané slovo, d značí daný dokument a D je množina všech dokumentů. Výslednou vahou slova je součin jeho tf a idf. Vektory dokumentů jsou pak tvořeny těmito váhami slov. Stejně, jako BoW tato metoda nezachovává informaci o pořadí slov. [18, 24]

2.3.2.3 Word2Vec

Na rozdíl od předchozích metod, Word2Vec nepřevádí do vektorové reprezentace dokumenty, ale jednotlivá slova. Získané vektory zachycují syntaktické a sémantické vztahy mezi slovy tak, že vektory slov s podobným významem jsou si blízké. Word2Vec je skupina modelů využívajících neuronové sítě k vytvoření vektorové reprezentace slov. Do této skupiny patří Continuous Bag-of-words Model (CBOW) a Skip-gram Model. Oba modely jsou trénovány pomocí stochastického gradientního sestupu, přičemž gradient je získán pomocí zpětného řetězení. [25]

Model CBOW funguje na následujícím principu. Každé slovo má na začátku unikátní vektor, namapovaný na sloupec matice W. Tyto vektory jsou v matici indexovány podle pozice daného slova ve slovníku. Zřetězení či průměr vektorů několika po sobě jdoucích slov v dokumentu (nazýváme kontext) se pak využívá k predikování následujícího slova. Cílem při trénování je maximalizace pravděpodobnosti, že následující slovo bude na základě daného kontextu správně predikováno. [26]

Skip-gram funguje na opačném principu. Predikuje určitý počet okolních slov v dokumentu pro jedno dané slovo. Při trénování tedy v každé iteraci vybereme textové okno a z tohoto okna náhodně zvolíme jedno slovo, na základě kterého se snažíme predikovat ostatní slova v daném okně. [26]

Word2Vec umožňuje zachytit informace o významu jednotlivých slov a jejich vzájemných vztazích. To je velká výhoda oproti výše uvedeným metodám. Abychom tyto informace mohli použít při klasifikaci, musíme je zakomponovat do vektorů dokumentů. Existuje několik způsobů, které k tomu můžeme použít. Vektory dokumentů můžeme získat například průměrováním vektorů slov, které se v dokumentu vyskytují. Další možností je využití Doc2Vec, který je rozšířením modelu Word2Vec a umožňuje získat také vektory dokumentů. [27]

2.3.2.4 Doc2Vec

Informace o tomto modelu jsme čerpali z článku [26].

Doc2Vec funguje na velmi podobném principu jako Word2Vec, umožňuje však získat vektory celých dokumentů. Vytváří vektory fixní velikosti pro dokumenty libovolné délky tak, že vektory podobných dokumentů jsou si blízké. Stejně, jako u modelu Word2Vec i zde jsou k dispozici dva různé algoritmy: Distributed memory (DM) a Distributed bag-of-words (DBOW).

Model DM funguje na podobném principu, jako výše popsaný model CBOW z Word2Vec. Při trénování však nevytváříme predikce následujícího slova pouze na základě vektorů okolních slov, ale využíváme také vektor dokumentu. Každý dokument má na začátku přidělen unikátní vektor fixní délky, který je v průběhu trénování postupně modifikován.

Model DBOW je založen na modelu Skip-gram z Word2Vec. Od původního modelu pro vytváření vektorů slov se liší pouze tím, že při trénování zohledňuje vektor dokumentu.

2.3.3 Redukce dimenzionality

Získané vektory dokumentů mívají stovky či tisíce příznaků. Takto vysoká dimenze prostoru příznaků má negativní vliv na efektivitu a úspěšnost klasifikátoru. Cílem redukce dimenzionlaity je snížení počtu příznaků při minimální ztrátě informací. Existují 2 různé přístupy, kterými tohoto výsledku můžeme docílit.

2.3.3.1 Feature selection

Při redukci dimenzionality pomocí feature selection vybíráme z prostoru příznaků takové příznaky, které jsou pro klasifikaci důležité (podle zvolené metriky). Typicky postupujeme následovně. Přiřadíme každému příznaku v daném dokumentu určitou váhu a následně vybereme k příznaků s nejvyššími vahami přes všechny dokumenty. Tyto váhy můžeme určit například na základě četnosti výskytů slova v dokumentu, tf-idf či informačního zisku (IG), který vypočteme podle vzorce:

$$\begin{aligned} \mathrm{IG}(t) &= H(c) \; - \; H(c \mid t) \\ &= - \sum_{i=1}^{m} \mathrm{P}(c_i) \, log(\mathrm{P}(c_i)) \\ &\quad + \mathrm{P}(t) \sum_{i=1}^{m} \mathrm{P}(c_i \mid t) \; \log(\mathrm{P}(c_i \mid t)) \\ &\quad + \mathrm{P}(\neg t) \sum_{i=1}^{m} \mathrm{P}(c_i \mid \neg t) \; \log(\mathrm{P}(c_i \mid \neg t)), \end{aligned} \tag{2.2}$$

kde t je daný token (slovo či n-gram), $c_i \in c$ je jedna z m kategorií, do kterých text klasifikujeme a $H(c)$ značí entropii těchto kategorií. [28, 29]

Dále se pro výběr k nejlepších příznaků využívá analýza rozptylu nazývaná ANOVA F-test. Pro výpočet skóre jednotlivých příznaků s využitím této metody potřebujeme nejdříve nalézt rozptyl hodnot daného příznaku mezi třídami (hodnotami vysvětlované proměnné), který vypočítáme podle vzorce:

$$\sigma^2_{mezi} = \frac{\sum_{i=1}^{M} n_i \, (\overline{x_i} - \overline{X})^2}{M - 1}, \tag{2.3}$$

kde M je počet různých tříd, n_i je počet záznamů příslušných dané třídě, $\overline{x_i}$ značí průměr hodnot příznaku x v i-té třídě a $\overline{X}$ značí střední hodnotu příznaku napříč všemi třídami. Dále spočítáme rozptyl uvnitř jednotlivých tříd následujícím způsobem:

$$\sigma^2_{uvnitř} = \frac{\sum_{i=1}^{M} \sum_{j=1}^{n_i} (x_{ij} - \overline{x_i})^2}{n - M}, \tag{2.4}$$

kde proměnná n značí celkový počet záznamů v datasetu. Výsledné skóre je následně vypočteno, jako podíl rozptylu σ^2_{mezi} ku $\sigma^2_{uvnitř}$. [30]

2.3.3.2 Feature transformation

Dalším možným přístupem pro snížení počtu příznaků je využití metod feature transformation. Tyto metody převádí prostor příznaků na prostor nižší dimenze za pomoci lineárních či nelineárních kombinací hodnot původních vektorů.

Metoda hlavních komponent

Nejznámější z těchto metod redukce dimenzionality je metoda hlavních komponent, zkráceně také PCA (z anglického názvu Principal Component Analysis). Jako hlavní komponenty zde označujeme nové příznaky vytvořené lineární kombinací příznaků původních. Hledání hlavních komponent můžeme chápat, jako hledání směrů, ve kterých mají data největší rozptyl. Prvním krokem k nalezení hlavních komponent je středování datasetu, které provádíme odečtením průměrné hodnoty příznaku od každé z jeho hodnot. Snažíme se nalézt projekci středovaného

datasetu na q rozměrný podprostor V s minimální kvadratickou chybou. Matice podprostoru V je tvořena prvními q vektory ortonormální báze. Tato báze je tvořena sestupně seřazenými vlastními vektory vlastních čísel výběrové varianční matice. Výběrová varianční matice je dána výrazem:

$$\frac{1}{N-1}X'^{T}X', \tag{2.5}$$

kde N je počet datových bodů a X' je středovaný dataset. Tato matice obsahuje na pozici (i,j) prvek, který je odhadem kovariance příznaků X_i a X_j. [31]

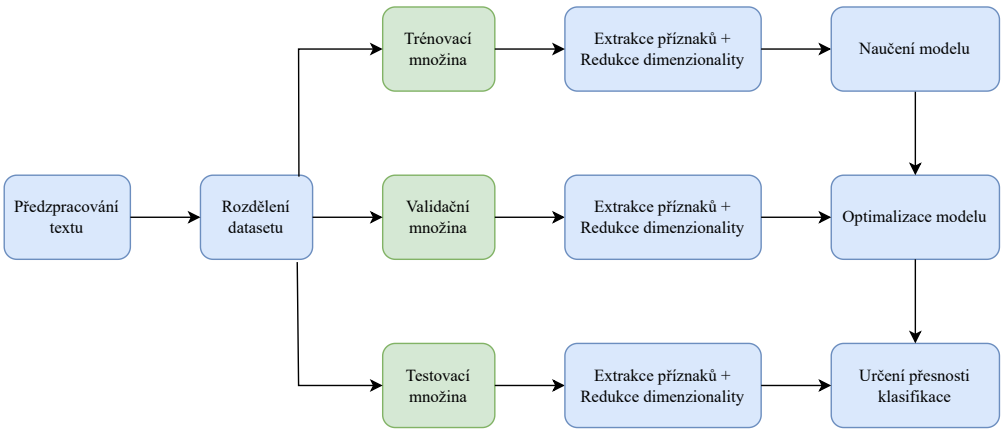
2.3.4 Modely pro klasifikaci textu

Strojové učení je jedním z odvětví umělé inteligence. V této práci jsme se zaměřili na modely supervizovaného učení, které se v průběhu trénování přizpůsobují datům z trénovací množiny a snaží se co nejlépe zachytit vztah mezi hodnotami příznaků a příslušnou kategorií (vysvětlovanou proměnnou). Na základě těchto vztahů pak vytvářejí predikce kategorií pro další data. Proces trénování modelu můžeme ovlivnit nastavením hyperparametrů. [10]

Ladění hyperparametrů

Při implementaci klasifikátoru je dataset typicky rozdělen na trénovací, validační a testovací množinu. Největší část dat by měla obsahovat trénovací množina, která slouží k naučení vybraného modelu. Při učení se klasifikátor přizpůsobuje datům tak, aby co nejlépe modeloval vztahy mezi příznaky a hodnotami vysvětlované proměnné. Validační množinu využíváme k ladění hyperparametrů, které nám umožňují ovlivnit tvar klasifikačního modelu. Při jejich ladění hledáme hodnoty hyperparametrů, které maximalizují přesnost klasifikace na validačních datech. Pro určení přesnosti výsledného modelu na neznámých datech se využívají data z testovací množiny. [32]

Obrázek 2.1 Diagram využití dat při konstrukci klasifikačního systému



Alternativně můžeme provést ladění hyperparametrů za pomoci cross-validace. Jedná se o metodu pro odhad klasifikační přesnosti modelu. Tento postup je výhodný při práci s malými datasety. Vzhledem k tomu, že zde nevytváříme validační množinu, můžeme využít větší část dat pro trénování modelu. Postup je však výpočetně náročný. Základní formou křížové validace je k-fold cross-validace, která funguje následovně. Dataset nejprve rozdělíme na trénovací a testovací množinu. Trénovací data dále rozdělíme do k podobně velkých, disjunktních podmnožin. Pro každou hodnotu hyperparametru provádíme k-krát trénování modelu. Pokaždé při tom vynecháme jednu

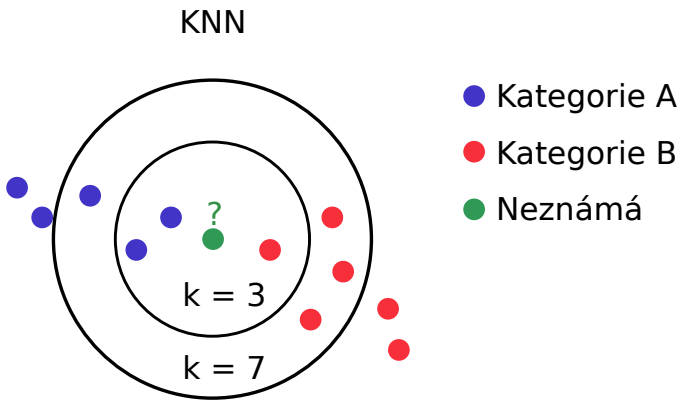
z k podmnožin, kterou použijeme, jako validační množinu a změříme na ní chybu klasifikace. Po naměření k chyb pro daný hyperparametr jejich hodnoty zprůměrujeme. Tyto průměrné chyby klasifikace pak můžeme porovnat a vybrat tak hyperparametry s nejlepšími výsledky. [33, 32]

2.3.4.1 K-Nearest Neighbors

Informace o tomto modelu byly čerpány z výukového materiálu [32].

Jako první si zde představíme model K-Nearest Neighbors, zkráceně KNN. Principem tohoto modelu je nalezení k dokumentů z trénovací množiny, které jsou podle zvolené metriky nejbližší dokumentu, který se snažíme zařadit do správné kategorie. Výslednou kategorií je pak ta, do které spadá největší počet z nalezených k dokumentů. Za naučený model je zde považována samotná množina trénovacích dat. Predikce jsou vytvářeny na základě vyhledávání podobných záznamů v této množině a fáze trénování modelu je tedy opomenuta. Vytváření predikcí tak bývá časově náročné. Pro každý predikovaný vzorek dat musíme spočítat vzdálenosti od všech záznamů v trénovací množině a tyto vzdálenosti porovnat. Tento proces můžeme urychlit indexací dat v trénovací množině.

Obrázek 2.2 Vizualizace algoritmu KNN [34]



Na obrázku 2.2 vidíme, jakým způsobem může nastavení parametru k ovlivnit výsledky predikcí. Pro k rovno 3 predikuje algoritmus kategorii A, zatímco při zvýšení hodnoty parametru k na 7 je predikována kategorie B.

Nastavením hyperparametru můžeme určit hodnotu k, tedy počet nejbližších sousedů, podle kterých budeme vytvářet predikce. Zvýšením hodnoty k můžeme zabránit přeučení modelu. Přeučením myslíme přílišné přizpůsobení datům z trénovací množiny, které snižuje přesnost predikce. Hyperparametry dále umožňují zvolit metriku pro výpočet vzdáleností. Nejčastěji využívanou metrikou je Eukleidovská vzdálenost:

$$\mathrm{d}_2(x, y) = \sqrt{\sum_{i=0}^{p-1} (x_i - y_i)^2}, \tag{2.6}$$

kde p je počet příznaků a x, y jsou body reprezentované jejich příznaky.

2.3.4.2 Naivní Bayesův klasifikátor

Zdrojem informací pro tuto část je [35].

Jedná se o klasifikační model založený na podmíněné pravděpodobnosti. Naivita spočívá v předpokladu nezávislosti příznaků pro fixní hodnotu vysvětlované proměnné. Při klasifikaci textu tedy předpokládáme, že výskyt slova v textu nezávisí na výskytu jakéhokoli jiného slova. Tento předpoklad často není splněný, kvůli používaným slovním spojením a frázím, které se v textu vyskytují. Naivní Bayesův klasifikátor přesto dosahuje dobrých výsledků. Nepřesnosti způsobené neplatným předpokladem často nemají vliv na výsledek klasifikace.

Snažíme se na základě trénovací množiny odhadnout pravděpodobnosti $\mathrm{P}(Y = y \mid X = x)$, pro všechny příznaky x z náhodného vektoru příznaků X a hodnoty y vysvětlované proměnné Y. Tyto pravděpodobnosti následně využijeme k predikci Y pro pozorované příznaky x. Tuto predikci nazýváme MAP odhad a provádíme ji nalezením nejpravděpodobnější hodnoty vysvětlované proměnné pro dané příznaky:

$$\widehat{Y} = \arg\max_{y \in Y} P(Y = y \mid X = x). \tag{2.7}$$

S využitím Bayesovy věty můžeme tento předpis převézt na následující vzorec.

$$\widehat{Y} = \arg\max_{y \in Y} P(X = x \mid Y = y)P(Y = y). \tag{2.8}$$

Předpoklad nezávislosti příznaků umožňuje rozložit pravděpodobnost $\mathrm{P}(X = x \mid Y = y)$ na součin podmíněných pravděpodobností jednotlivých příznaků pro fixní hodnotu vysvětlované proměnné. Dochází tak k separaci příznaků, která pomáhá předcházet problémům s vysokou dimenzí. Výsledný MAP odhad pak vypadá takto:

$$\widehat{Y} = \arg\max_{y \in Y} \prod_{i=1}^{p} P(X_i = x_i \mid Y = y)P(Y = y). \tag{2.9}$$

Při odhadování pravděpodobnosti $\mathrm{P}(X = x \mid Y = y)$ záleží na zvoleném způsobu extrakce příznaků. Pokud pro reprezentaci textů zvolíme například model bag-of-words, můžeme pro výpočet této pravděpodobnosti použít následující vzorec:

$$P(X = x \mid Y = y) = \frac{n!}{\prod_{i=1}^{D} x_i!} \prod_{i=1}^{D} p_{i,y}^{x_i}, \tag{2.10}$$

kde n je počet slov v daném dokumentu, x_i je počet výskytů i-tého slova ze slovníku D v daném dokumentu a $p_{i,y}$ označuje pravděpodobnost, že náhodné slovo z dokumentu, spadajícího do kategorie y je i-tým slovem ze slovníku D. Odhad pravděpodobnosti $p_{i,y}$ s využitím add-one smoothing získáme podle vzorce:

$$\widehat{p_{i,y}} = \frac{N_{i,y} \; + \; 1}{N_y \; + \; D}, \tag{2.11}$$

kde $N_{i,y}$ značí počet výskytů i-tého slova ze slovníku D ve všech dokumentech z kategorie y. N_y je počet všech slov v dokumentech kategorie y.

2.3.4.3 Logistická regrese

V této části čerpáme z výukového materiálu [36].

Logistická regrese je klasifikační model založený na předpokladu, že můžeme chování dat zachytit pomocí sigmoidní křivky. Fungování tohoto modelu pro jednoduchost vysvětlíme na případu binární klasifikace. Příkladem binární klasifikace je rozhodování, zda se jedná o extremistický text či nikoli. Vysvětlovaná proměnná Y zde nabývá hodnot 0 a 1. Stejně, jako u lineární regrese zde rozhodujeme na základě lineární kombinace příznaků ve tvaru:

$$w_0 + w_1 x_1 + ... + w_n x_n, \tag{2.12}$$

kde $x_1, ..., x_n$ jsou hodnoty příznaků a $w_0, ..., w_n$ jsou neznámé koeficienty. Hodnoty koeficientů nastavujeme ve fázi učení modelu na základě trénovacích dat. Pro nalezení těchto hodnot se využívá maximálně věrohodný odhad (MLE). Abychom mohli za pomoci lineárního výrazu 2.12 rozhodnout, zda je hodnota Y pro dané příznaky 1 či 0, převedeme nejprve tento binární problém na problém spojitý. Při logistické regresi toho docílíme tak, že namísto predikce hodnot vysvětlované proměnné predikuje pravděpodobnost, že vysvětlovaná proměnná Y nabývá hodnoty 1. Pravděpodobnost $P(Y = 1)$ závisí na hodnotách příznaků $x = (x_1, ..., x_n.)$ a koeficientů $w = (w_0, w_1, ..., w_n)$, kde n je počet příznaků. Hledanou pravděpodobnost značíme:

$$P(Y = 1 \mid x, w). \tag{2.13}$$

Při binární klasifikaci můžeme pravděpodobnost opačného jevu snadno odvodit, jako:

$$P(Y = 0 \mid x, w) = 1 - P(Y = 1 \mid x, w). \tag{2.14}$$

Dále musíme zajistit, aby výsledek výrazu 2.12 patřil do intervalu [0,1]. Tedy aby dával smysl jakožto míra pravděpodobnosti. Toho docílíme dosazením výrazu 2.12 do funkce s oborem hodnot v intervalu [0,1]. Obvykle se k tomuto účelu používá sigmoida, která je speciálním případem logistické funkce. Po dosazení tedy dostáváme:

$$P(Y = 1 \mid x, w) = \frac{e^{w^T x}}{1 + e^{w^T x}}. \tag{2.15}$$

Tuto pravděpodobnost pak využijeme k predikci hodnot vysvětlované proměnné. Pokud pro dané příznaky vyjde $P(Y = 1 \mid x, w) > \frac{1}{2}$, je výsledkem predikce $Y = 1$, v opačném případě řekneme, že $Y = 0$.

2.3.4.4 Support vector machine

Informace pro popis tohoto modelu byly čerpány ze zdrojů [37, 24]

Support vector machine, zkráceně SVM je často využívaným modelem při klasifikaci textu. Používá se pro binární klasifikaci, ale existují i rozšíření tohoto modelu pro klasifikaci do více tříd či pro regresi. Klasifikátor se snaží oddělit trénovací data z různých kategorií, reprezentovaná jako body v prostoru, pomocí nadroviny. Abychom mohli takto oddělit jednotlivé kategorie, musí platit předpoklad, že jsou data lineárně separabilní.

Lineárně separabilní data

Uvažme problém binární klasifikace lineárně separabilních dat. V tomto případě můžeme použít maximal margin classifier, který je nejjednodušším modelem z rodiny SVM. Data z trénovací množiny reprezentujeme, jako body v prostoru příznaků. Hledáme pak takovou nadrovinu oddělující data z různých kategorií, která maximalizuje margin.

Na obrázku 2.3 vidíme, že margin je vzdálenost, mezi dvěma paralelními nadrovinami, které prochází nejbližšími datovými body z trénovací množiny (support vectory). Tyto hraniční nadroviny můžeme popsat rovnicemi:

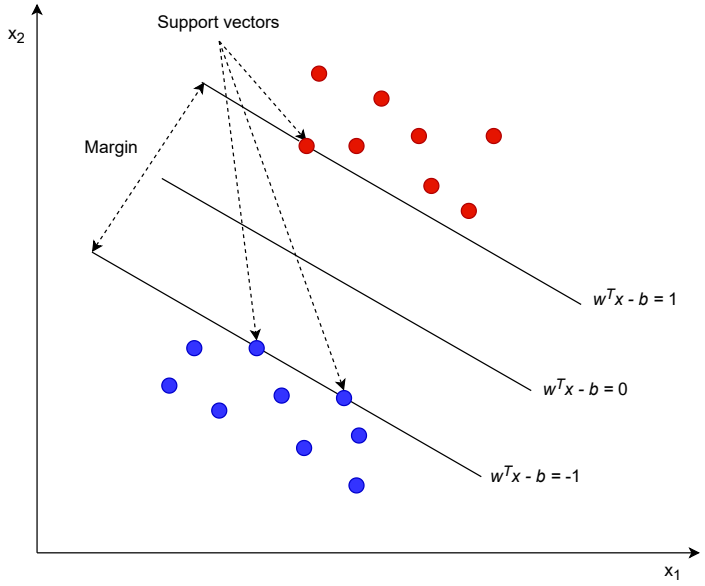
$$w^T x - b = 1, \;\; w^T x - b = \text{-}1, \tag{2.16}$$

kde x je vektor příznaků datového bodu, který leží na dané rovině, w je normálový vektor roviny a b je reálné číslo určující její posun. Hledaná nadrovina, která slouží ke klasifikaci dat do daných kategorií pak leží uprostřed, mezi nalezenými hraničními nadrovinami a má tvar:

$$(w^T x - b) = 0. \tag{2.17}$$

Vzdálenost mezi hraničními nadrovinami vypočítáme podle vzorce:

Obrázek 2.3 Model SVM v prostoru příznaků dimenze 2 [24]



$$margin = \frac{2}{\|w\|}. \tag{2.18}$$

Čím vyšší je margin, tím přesnější výsledky můžeme očekávat při následné klasifikaci dat. Ze vzorce 2.18 vidíme, že abychom margin maximalizovali, musíme minimalizovat normu vektoru w. Předpokládejme, že vysvětlovaná proměnná nabývá hodnot 1 a -1. Maximalizace marginu docílíme nalezením nejmenšího možného $\|w\|$, které pro všechna trénovací data i splňuje:

$$y_i \left(w^T x_i - b\right) \geq 1. \tag{2.19}$$

Při vytváření predikcí pracujeme s prostorem rozděleným na dvě části pomocí nalezené nadroviny. Pro zadané parametry x určíme, ve které části prostoru se daný datový bod nachází. Podle toho mu pak přiřadíme odpovídající kategorii.

Lineárně neseparabilní data

Data, která potřebujeme klasifikovat často nesplňují předpoklad lineární separability. Tento problém se dá řešit takzvaným kernelovým trikem, při kterém mapujeme datové body do prostoru vyšší dimenze, kde již můžeme data oddělit pomocí nadroviny. Důležité je zvolit vhodnou kernelovou funkci pro mapování dat. [38]

Dalším možným řešením tohoto problému je využití takzvaného soft-margin klasifikátoru. Ten funguje na podobném principu, jako výše zmíněný maximal margin classifier s tím rozdílem, že toleruje malé množství chybně klasifikovaných trénovacích data. Díky tomu můžeme model použít i na data obsahující určité procento chyb či anomálií. Model se snaží minimalizovat součet vzdáleností chybně klasifikovaných bodů od příslušné hraniční nadroviny. Pro každý datový bod z trénovacího datasetu definujeme veličinu ξ, která má hodnotu 0, pokud je daný bod klasifikován správně. V opačném případě má hodnotu odpovídající vzdálenosti tohoto bodu od příslušné nadroviny $w^T x - b = \pm 1$. Můžeme ji tedy popsat pro i-tý datový bod následujícím vztahem:

$$\xi_i = \max\left\{0,\, 1 - y_i \left(w^T x_i - b\right)\right\}. \tag{2.20}$$

Hodnotu w s nejmenší možnou chybou klasifikace pak hledáme, jako minimum výrazu

$$\frac{1}{N}\sum_{i=1}^{N}\xi_i + C\|w\|^2, \tag{2.21}$$

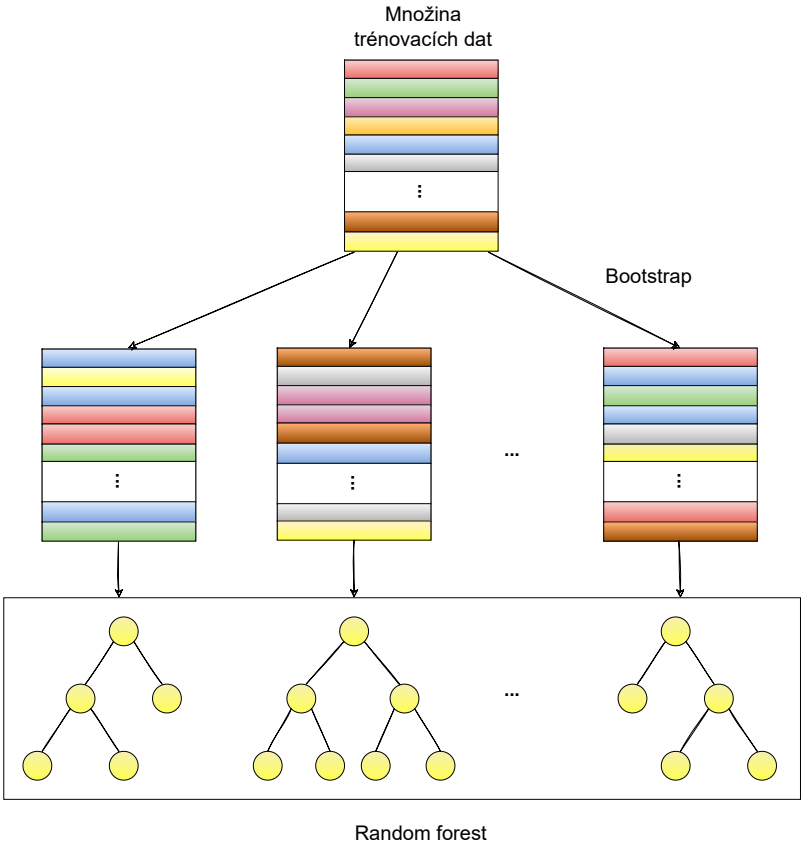
kde N je počet dat v trénovací množině a C je parametr určující jak moc mají být chyby penalizovány.

2.3.4.5 Random forest

Informace o tomto modelu jsme čerpali ze zdrojů [39, 40, 16]

Random forest (česky také Náhodný les) patří do kategorie ensamble modelů a dá se využít pro klasifikační i regresní úlohy. Ensamble modely fungují na principu vytváření více jednoduchých modelů, které při následném vytváření predikcí o výsledku hlasují. Základní myšlenkou tohoto přístupu je, že kombinací výsledků více slabých modelů můžeme docílit přesnějších predikcí. Random forest je tvořen určitým počtem rozhodovacích stromů, obvykle malé hloubky. Důležitá je zde rozmanitost dílčích stromů, která významně ovlivňuje úspěšnost klasifikace.

Obrázek 2.4 Proces vytvoření náhodného lesa



Rozhodovací stromy typicky vytváříme pomocí rekurzivního dělení dat. Z množiny příznaků vždy vybereme ten, se kterým při rozdělení dat z daného uzlu dosáhneme největšího snížení neuspořádanosti dat. Na takto rozdělená data pak aplikujeme stejný postup. Dělení dat (neboli split) provádíme, dokud nenastane některá z ukončovacích podmínek daných hyperparametry. Pro určení nejlepšího splitu se dá využít Entropie, která je dána následujícím vzorcem:

$$H(D) = -\sum_{i=0}^{k-1} p_i \log p_i, \tag{2.22}$$

kde p_i je počet záznamů i-té kategorie v dané množině dat D a k je počet různých kategorií. Pro určení nejlepšího splitu se využívá také Gini index:

$$GI(D) = \sum_{i=0}^{k-1} p_i \, (1 - p_i). \tag{2.23}$$

Při implementaci tohoto modelu musíme nejprve vytvořit z trénovacích dat tolik datasetů, kolik chceme mít v lese rozhodovacích stromů. K tomu se využívá metoda bootstrap. Ta vytváří z trénovací množiny dat nové datasety stejné délky, pomocí náhodného výběru záznamů s opakováním. Na těchto datasetech pak trénujeme jednotlivé rozhodovací stromy. Počet stromů v lese a jejich tvar určujeme nastavením příslušných hyperparametrů.

Různorodost dílčích stromů je dána tím, že každý z nich trénujeme na jiném datasetu, díky využití metody bootstrap. Dále se běžně používá omezením počtu příznaků, ze kterých se při trénování stromu vybírá ten s nejlepším splitem. Daný počet příznaků je pak náhodně vybírán z množiny příznaků pro každý split zvlášť.

Klasifikace pak probíhá následujícím způsobem. Každý strom necháme sestrojit vlastní predikci vysvětlované proměnné pro zadané hodnoty příznaků. Jako výslednou predikci modelu vybereme tu, kterou určil největší počet dílčích stromů.

2.3.4.6 AdaBoost

Použitým zdrojem informací pro tuto část práce byl studijní materiál [40].

Jedná se opět o ensamble model tvořený kolekcí více slabých modelů. Těmito modely mohou být například rozhodovací stromy, tak jako u výše zmíněného Random forest. AdaBoost však vytváří dílčí klasifikátory postupně tak, že každý model je závislý na modelech předchozích. Tento způsob vytváření ensamble modelů se nazývá Boosting. Jeho cílem je minimalizace trénovací chyby modelů s využitím vážených dat při jejich trénování. AdaBoost byl původně navržen pro binární klasifikaci, existují však i rozšíření tohoto algoritmu pro klasifikaci do více kategorií či pro regresi. Při popisu tohoto modelu se omezíme na použití pro binární klasifikaci.

AdaBoost na rozdíl od Random forest využívá ke konstrukci dílčích modelů vážená data. Na začátku mají všechna data stejnou váhu w, tedy $w_i = \frac{1}{n}$, kde n je počet záznamů v trénovací množině. Po vytvoření prvního modelu zjistíme, která trénovací data jím byla zařazena do špatné kategorie. Těmto datům následně zvýšíme váhu, čímž docílíme toho, že jim bude věnována zvýšená pozornost při trénování dalších modelů. Zvýšení vah můžeme provést podle vzorce:

$$w_i \leftarrow \frac{1 - e^{(m)}}{e^{(m)}} w_i, \tag{2.24}$$

kde m je index modelu a $e^{(m)} \in (0, \frac{1}{2})$ je součet vah špatně klasifikovaných datových bodů daným modelem. Abychom mohli tento vzorec použít, musí být hodnota $e^{(m)}$ menší, než $\frac{1}{2}$), jinak by došlo k nežádoucímu snížení vah. Po každém zvýšení musíme váhy normalizovat tak, aby byl jejich součet roven jedné. Při trénování každého dalšího klasifikátoru tedy využíváme takto upravené váhy a následně je znovu modifikujeme podle dosažených výsledků. Ve vytváření modelů pokračujeme, dokud nedosáhneme dostatečně nízké chyby $e^{(m)}$, nebo dokud nevytvoříme stanovený počet modelů.

Každému z dílčích modelů $T^{(m)}$je přiděleno skóre $s_T^{(m)}$, které využijeme při vytváření predikcí.

$$s_T^{(m)} = learning\_rate \times \log \frac{1 - e^{(m)}}{e^{(m)}} \tag{2.25}$$

Hodnotu learning_rate nastavujeme pomocí hyperparametru. Čím vyšší je jeho hodnota, tím vyšší váhu mají jednotlivé modely. Predikce vytváříme tak, že porovnáme součet $s_T^{(m)}$ všech modelů, které pro zadané parametry predikovaly hodnotu Y=1 se součtem $s_T^{(m)}$ modelů, které určily Y=0. Jako výslednou predikci určíme tu, která dosáhla vyššího celkového skóre.

2.3.5 Vyhodnocení přesnosti klasifikace

Použitý zdroj informací: [29]

Po dokončení implementace klasifikátoru následuje fáze, při které zjišťujeme úspěšnost klasifikace na testovacích datech. Testovací data nebyla použita při učení modelu a neměla vliv na ladění hyperparametrů. Dají se tedy použít k odhadu přesnosti klasifikace neznámých dat. Při měření úspěšnosti necháme klasifikátor vytvořit predikce vysvětlovaných proměnných pro testovací data a porovnáme je s jejich správnými kategoriemi. Pro vyjádření míry úspěšnosti se dají použít různé metriky. Nejpoužívanějšími metrikami jsou accuracy, precision a recall. Pro pochopení těchto metrik musíme nejdříve definovat následující termíny.

- True positive (TP) je počet správně zařazených dokumentů do dané kategorie.
- False positive (FP) je počet nesprávně zařazených dokumentů do dané kategorie.
- True negative (TN) je počet dokumentů, které do kategorie nepatří a nejsou do ní přiřazeny.
- false negative (FN) je počet dokumentů, které do dané kategorie patří, ale nejsou do ní přiřazeny.

2.3.5.1 Accuracy

Accuracy je nejvíce intuitivní a nejčastěji používanou metrikou při hodnocení úspěšnosti klasifikace. Vyjadřuje procento správně klasifikovaných dokumentů do dané kategorie. Do češtiny ji můžeme přeložit jako správnost. Vzorec pro její výpočet u kategorie c_i vypadá následovně:

$$Acc_i = \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}. \tag{2.26}$$

2.3.5.2 Precision

Precision se do češtiny překládá, jako přesnost. Říká nám, jaký podíl dokumentů zařazených do dané kategorie je zde zařazených správně. Vyjadřuje tedy schopnost klasifikátoru zařadit dokument do správné kategorie. Pro kategorii c_i ji vypočítáme podle vzorce:

$$Prec_i = \frac{TP_i}{TP_i + FP_i}. \tag{2.27}$$

2.3.5.3 Recall

V češtině tuto metriku označujeme, jako úplnost. Udává procento dokumentů patřících do dané kategorie jsme byli schopni identifikovat. Pro výpočet používáme vzorec:

$$Rec_i = \frac{TP_i}{TP_i + FN_i}. \tag{2.28}$$

2.3.5.4 F1 score

Tato metrika je harmonickým průměrem precission a recall. [10] Vypočítáme ji tedy následovně:

$$F1_i = \frac{2\,(Rec_i * Prec_i)}{Rec_i + Prec_i}. \tag{2.29}$$

2.4 Přehled související literatury

Na závěr rešeršní části práce si ukážeme využití výše popsaných metod klasifikace textu pro detekci extremistických textů. V sekci *Vývoj metod detekce extremismu v online prostoru* jsme si již představili několik studií, využívajících různé přístupy k identifikaci extremistického obsahu na internetu. V této části se zaměříme na současné odborné práce, které využívají metody supervizovaného strojového učení. Ukážeme na jich, jaké metody byly k řešení problému použity a jakých výsledků bylo s jejich využitím dosaženo.

Jako první si zde představíme článek [41] zaměřený na detekci radikalizace a extremismu s využitím metod NLP. Konkrétně zde byly využity metody pro tokenizaci textu, stemming a odstranění stopslov, ve fázi předzpracování textových dokumentů. Extrakce příznaků spočívala ve využití dvou různých metod: tf-idf a word2vec. Pro samotnou klasifikaci textů byly použity dva ensamble modely: Random forest a Gradient boosting. Ve fázi experimentů pak byla zkoumána úspěšnost klasifikace těchto modelů v závislosti na zvolené metodě pro extrakci příznaků. Úspěšnost klasifikace byla měřena pomocí následujících metrik: accuracy, precision, recall a F1 score. Výsledkem těchto experimentů bylo zjištění, že oba modely dosáhly vyšší úspěšnosti při klasifikaci na základě vektorové reprezentace získané metodou word2vec. Model Gradient boosting dosáhl v obou případech lepší úspěšnosti než Random forest. Rozdíl úspěšnosti těchto modelů se u všech použitých metrik pohyboval okolo dvou procent.

Jako další příklad zde uvádíme článek [42] zaměřený na identifikaci extremistických tweetů. Dataset byl sestaven z tweetů, rozdělených do tří kategorií: extremistické pro-Tálibán, extremistické pro-Afghánistán a neutrální. Po předzpracování textových dokumentů pomocí metod NLP byla data převedena do vektorové podoby s využitím metody tf-idf pro unigramy a bigramy. Po snížení počtu příznaků s využitím metody PCA byla provedena Explorační analýza dat (Exploratory Data Analysis). Tato metoda umožňuje vizualizovat skryté vzory chování dat a vede tak k jejich lepšímu pochopení. Ke klasifikaci textů bylo použito několik klasifikačních modelů: naivní Bayesův klasifikátor, SVM, rozhodovací stromy, KNN, a ensamble modely využívající bagging a boosting. Ve fázi experimentů se pak porovnávala úspěšnost těchto klasifikačních modelů v závislosti na zvolené množině příznaků (unigramy, bigramy a příznaky redukované metodou PCA). Jejich výsledky jsou znázorněny v následující tabulce.

Tabulka 2.2 Výsledky experimentů z článku [42]

Classification Algorithms	Unigram				Bigram				Reduced Feature Set (Using PCA)			
	Accuracy	Precision	Recall	F-Score	Accuracy	Precision	Recall	F-Score	Accuracy	Precision	Recall	F-Score
Naïve Bayes'	**83.16**	**82.81**	**80.2**	81.48	81.05	79.73	80.84	80.28	68.06	67.87	67.70	67.78
SVM	80.73	80.65	77.21	78.89	**84.71**	**85.18**	**83.15**	84.15	74.10	75.00	73.21	74.09
Decision Tree	76.06	79.56	69.28	74.06	77.43	80.22	71.34	75.52	70.70	76.17	68.77	72.28
Random Forest	70.7	74.04	70.04	71.98	69.56	84.70	50.87	63.56	**76.00**	**78.77**	**74.69**	76.68
KNN	74.78	73.05	72.82	72.93	73.87	70.34	66.29	68.25	72.55	73.85	73.33	73.59
Bagging	83.12	82.21	81.01	81.61	83.66	81.89	80.01	80.94	70.17	73.91	68.44	71.07
Boosting	81.81	80.30	78.56	79.42	84.43	86.29	82.76	84.49	72.52	73.39	71.43	72.40

Z tabulky vyplývá, že nejvyšší přesnosti při klasifikaci unigramů (1-gramů) dosáhl naivní Bayesův klasifikátor. Při klasifikaci bigramových příznaků dosáhl nejlepších výsledků model SVM. Random forest dosáhl nejvyšší přesnosti na příznacích po redukci PCA. Zajímavé je, že se přesnost klasifikace po redukci dimenzionality u většiny modelů zhoršila. Metoda PCA tedy způsobila ztrátu informací, které byly pro klasifikaci důležité.

Nakonec zde zmíníme článek [43], zaměřený na rozpoznávání jihadistických tweetů. Autoři pracovali s tweety rozdělenými do tří různých datasetů: pro-ISIS, proti-ISIS a neutrální. Při trénování klasifikačních modelů byly použity tři různé druhy příznaků.

- Stylometrické příznaky zahrnovaly například informace o četnostech nejčastěji se vyskytujících slov či bigramů, často používaných hashtagů a interpunkčních znamének.
- Časové příznaky obsahovaly informace o čase zveřejnění daného tweetu.

- Poslední typ příznaků zahrnoval informace o sentimentu daného textu. Tyto příznaky mohly nabývat hodnot: velmi negativní, negativní, neutrální, pozitivní a velmi pozitivní.

Po převedení textů do vektorové podoby byly odstraněny příznaky, které nepřinášely dostatečně vysoký informační zisk. Použity byly tři klasifikační modely: SVM, naivní Bayesův klasifikátor a AdaBoost. Ve fázi experimentů byly porovnávány výsledky klasifikace těchto modelů na datech z různých datasetů. Nejlepších výsledků bylo dosaženo při klasifikaci dat z neutrálního a pro-ISIS datasetu. Experimenty využívající proti-ISIS dataset ukázaly, že tento způsob klasifikace je pro modely podstatně náročnější. Nejvyšší accuracy dosáhl ve všech případech model AdaBoost s přesností klasifikace 99,5 - 100 %.

Kapitola 3
Praktická část

V této části práce nejprve uvádíme vybrané metody pro předzpracování, extrakci příznaků a klasifikaci textu, a dále popisujeme nástroje použité při implementaci. Tato kapitola je rozdělena do tří hlavních částí. První z nich se zabývá prvotními experimenty, které byly provedeny na provizorním datasetu obsahujícím anglické extremistické a neutrální texty. Cílem těchto experimentů bylo základní porovnání metod pro extrakci příznaků a použitých klasifikačních modelů. Na tuto část navazujeme při implementaci experimentů v trénovacím programu.

Další část je věnována představení trénovacího programu a popisu jeho implementace. Trénovací program, je podstatnou součástí této práce. Umožňuje vytváření a ukládání natrénovaných klasifikátorů, které jsou využívány v klasifikačním systému. Trénovací program zároveň provádí experimenty v rámci kterých zkoumáme vliv využití různých metod předzpracování textu a extrakce příznaků na úspěšnost klasifikace. Podle výsledků těchto experimentů jsou vybrány vhodné kombinace metod pro klasifikaci textu ve výsledném klasifikačním systému.

Na závěr je uveden popis klasifikačního systému. Tato část práce je velmi stručná, jelikož vytvořený klasifikační systém funguje na velice jednoduchém principu a pracuje s již natrénovanými klasifikátory.

3.1 Vybrané metody

V této části práce uvádíme zvolené metody, které využíváme při provádění experimentů. Při výběru vhodných metod vycházíme z poznatků získaných při rešerši související literatury. Popis využitých nástrojů implementujících tyto metody je uveden v následující sekci s názvem „Použité nástroje“ 3.2.

3.1.1 Předzpracování textu

Ve fázi předzpracování textu provádíme typické operace pro odstranění přebytečných znaků a slov, která nemají pro klasifikaci význam. Konkrétně provádíme odstranění interpunkce a speciálních znaků, převedení textu na malá písmena, tokenizaci a odstranění stopslov. Tyto metody používáme stejným způsobem při předzpracování textů v anglickém i českém jazyce. Liší se pouze množina stopslov, které z textů odstraňujeme.

Rozdíl nastává při převádění slov do základního tvaru, jelikož metody pro lemmatizaci a stemming slov jsou závislé na konkrétním jazyce. Při předzpracování anglických textů v prvotních experimentech využíváme metodu pro stemming slov z knihovny NLTK. Při provádění navazujících experimentů s českými texty porovnáváme vliv lemmatizace a stemmingu s využitím

dvou volně dostupných nástrojů pro český jazyk. Pro stemming českých slov využíváme modul czech_stemmer a pro lemmatizaci slov knihovnu simplemma.

3.1.2 Extrakce příznaků

Zvolení vhodné metody pro extrakci příznaků je klíčové pro dosažení vyšší přesnosti při detekci extremistických textů. Přestože se vlivem použití různých metod pro extrakci příznaků na přesnost klasifikace textu zabývá velké množství publikací není zřejmé, která z těchto metod dokáže poskytnout nejlepší výsledky v kombinaci s klasifikačním modelem. Záleží zejména na velikosti množiny trénovacích dat, způsobu předzpracování textu, délce textových dokumentů a rozmanitosti slov, které se v textech vyskytují. Budeme proto ve fázi experimentů zkoumat několik různých metod a následně vybereme tu, která dosáhne nejlepších výsledků. Konkrétně pracujeme s modely bag-of-words, tf-idf a Doc2Vec, které se při klasifikaci textu často využívají.

3.1.3 Klasifikační modely

Stejně, jako u výše zmíněných metod pro extrakci příznaků nemůžeme určit, který z modelů strojového učení je pro daný problém nejlepší volbou. V této práci budeme pracovat s modely SVM a Random forest. Oba modely byly již dříve využity pro detekci extremismu v odborných publikacích a dosáhly relativně dobrých výsledků. Zejména model SVM je v související literatuře často zmiňován. Model Random forest byl vybrán s očekáváním vyšší robustnosti a odolnosti vůči přeučení. 2.4.

3.2 Použité nástroje

Při implementaci všech tří dílčích programů vytvořených v rámci této práce využíváme jazyk Python. Python je vhodný pro klasifikaci textu především díky velkému množství dostupných nástrojů, které usnadňují práci s přirozeným jazykem, převod textu do vektorové reprezentace a používání modelů strojového učení. V této části práce představíme konkrétní knihovny a nástroje, které byly při implementaci použity.

3.2.1 NLTK

NLTK (Natural Language Toolkit) je sada knihoven a programů určených pro zpracování přirozeného jazyka v Pythonu. Nástroje z NLTK jsou určeny především pro práci s anglickým jazykem. Níže jsou popsány konkrétní nástroje, které využíváme pro předzpracování textových dokumentů. [44]

- **PorterStemmer**: Třída pro stemming anglických slov založená na originálním Porter stemming algoritmu publikovaném v roce 1980 [45]. Originální algoritmus využívá řadu pravidel pro odstranění přípon a koncovek z anglických slov. PorterStemmer umožňuje nastavení jednoho ze tří módů. Jedním z nich je právě originální algoritmus. Další dva módy obsahují rozšíření tohoto algoritmu.[46]

- **word_tokenize**: Metoda, která vrací tokenizovanou kopii textu, který dostala na vstupu. Tato metoda odděluje od textu interpunkční znaménka a speciální znaky a následně rozděluje text na tokeny podle bílých znaků. [47]

3.2.2 Scikit-learn

Scikit-learn je knihovna nabízející širokou řadu state-of-the-art algoritmů strojového učení. Umožňuje snadnou aplikaci těchto modelů na problémy supervizovaného i nesupervizovaného učení. Níže jsou popsány nástroje, které z této knihovny využíváme při extrakci příznaků, ladění hyperparametrů a klasifikaci textů pomocí modelů Random forest a SVM. [48]

- **RandomForestClassifier**: Třída implementující klasifikační model Random forest. Obsahuje metody pro trénování modelu, vytváření predikcí, získání nastavených hyperparametrů a další. [49]
- **SVC**: Třída pro klasifikaci pomocí modelu Support vector machine. Obsahuje stejné metody, jako RandomForestClassifier. [50]
- **CountVectorizer**: Třída implementující model bag-of-words. Převádí tedy kolekci textových dokumentů na matici obsahující počty výskytů jednotlivých slov či n-gramů v dokumentech. Umožňuje také nastavení různých parametrů, kterými můžeme například omezit velikost vygenerovaného slovníku či ignorování tokenů s příliš nízkým či vysokým počtem výskytů. [51]
- **TfidfVectorizer**: Třída pro převod kolekce textových dokumentů do podoby matice tf-idf příznaků. Jedná se o ekvivalent použití metody CountVectorizer s následnou transformací příznaků do reprezentace tf-idf. [52]
- **GridSearchCV**: Třída využívající k-fold cross validaci pro ladění hyperparametrů klasifikačních či regresních modelů strojového učení. Generuje všechny možné kombinace zadaných hodnot laděných parametrů a hledá jejich nejlepší kombinaci podle zadané metriky. [53]
- **ParameterGrid**: Třída pro generování všech možných kombinací zadaných hodnot hyperparametrů. [53]
- **train_test_split**: Funkce pro náhodné rozdělení datasetu na dvě množiny. Umožňuje nastavení velikosti výsledných množin a případné potlačení randomizace pro získání stejného rozdělení při opakovaném volání. [54]
- **SelectKBest**: Třída pro selekci příznaků. Ze zadané množiny příznaků vybírá k příznaků s nejvyšším skóre. Pro přiřazení skóre jednotlivým příznakům se defaultně využívá metoda analýzy rozptylu (ANOVA). [55]
- **metrics**: Modul obsahující mimo jiné funkce pro vyhodnocení úspěšnosti klasifikace či regrese podle různých metrik. [56]

3.2.3 Gensim

Gensim je knihovna používaná zejména pro reprezentaci dokumentů pomocí sémantických vektorů. Algoritmy z této knihovny, jako například Doc2Vec či FastText, využívají pro zachycení sémantické struktury dokumentů metody nesupervizovaného strojového učení. Z této knihovny využíváme pouze model **Doc2Vec**, jehož princip je popsán v rešeršní části práce 2.3.2.4. [57]

3.2.4 Simplemma

Simplemma je knihovna umožňující snadnou lemmatizaci slov, která v současné době částečně či plně podporuje 38 různých jazyků, mezi které patří také čeština. Dokáže lemmatizovat série tokenů či celé texty díky integrovanému tokenizeru. [58]

3.2.5 czech_stemmer

Pro stemming českých slov používáme upravený modul czech_stemmer dostupný zde: [59]. Jedná se o reimplementaci algoritmů pro stemming českých slov, které představili autoři Ljiljana Dolamic a Jacques Savoy v článku [60]. V tomto článku byly navrženy algoritmy pro lehký a agresivní stemming. Podle definovaných pravidel jsou především z podstatných a přídavných jmen odstraňovány přípony a koncovky. Algoritmy se nezabývají úpravou méně častých slovesných tvarů, ani odstraňováním předpon.

Použitý modul czech_stemmer vytvořil Luís Gomes, jako reimplementaci výše popsaných stemmovacích algoritmů v jazyce Python. Modul je navržen primárně jako konzolová aplikace, dá se však využít i pro předzpracování textů v programech implementovaných v Pythonu. Nástroj umožňuje zvolit pomocí parametru, zda chceme provádět pouze lehký či agresivní stemming.

Provedli jsme úpravy tohoto modulu, zahrnující především odstranění funkce main, která sloužila ke komunikaci s terminálem. Dále jsme odstranili několik přebytečných operací, které sloužily k zachování velikosti písmen.

3.2.6 Joblib

Z této sady nástrojů používáme funkce joblib.dump a joblib.load, které umožňují ukládání modelů a datových struktur do souborů ve formátu PKL a jejich opětovné načítání. Díky tomuto nástroji můžeme ve výsledném systému pracovat s před-trénovanými klasifikátory, které pouze načítáme ze souborů. Jedná se o obdobu modulu pickle optimalizovanou pro efektivní práci s objekty obsahujícími velké množství dat. [61]

3.2.7 pandas

Pandas je knihovna poskytující datové struktury a nástroje pro analýzu dat. Využíváme ji pro načítání datasetů, které jsou uloženy v CSV souborech a pro následnou manipulaci s načtenými daty. [62]

3.3 Prvotní experimenty

V této části práce popíšeme implementaci jednoduchého programu pro porovnání úspěšnosti klasifikace modelů SVM a Random forest při využití různých metod pro extrakci příznaků. U porovnávaných metod a klasifikátorů využíváme defaultní hodnoty hyperparametrů, jejich ladění se budeme věnovat až v navazujících experimentech. Výstupem programu je přehled dosažených hodnot accuracy precision a recall jednotlivých modelů v kombinaci s příslušnou metodou extrakce příznaků.

Tato prvotní implementace sloužila k rozpracování experimentů s vybranými metodami pro klasifikaci textu, zatímco nebyl k dispozici dataset s extremistickými texty v češtině. Pracujeme zde s provizorním datasetem obsahujícím krátké anglické texty rozdělené do dvou kategorií: extremistické a ne-extremistické. Z této základní implementace budeme vycházet při implementaci experimentů v trénovacím programu.

3.3.1 Tvorba provizorního datasetu

Provizorní dataset jsme vytvořili v Jupyter Notebooku úpravou a následným spojením dvou různých, volně dostupných datasetů. Výsledný dataset uložen ve formátu CSV a obsahuje 190 extremistických a 752 ne-extremistických záznamů. Tato nevyváženost odpovídá relativně nízkému výskytu extremistických textů v reálném světě. Nejedná se o příliš velký vzorek dat, ale pro účely orientačního porovnání použitých metod by měl být dostačující.

Prvním datasetem, který jsme zde použili je dataset ISIS_Seed_Complete.csv, získaný z úložiště Zenodo [63]. Tento dataset obsahuje 200 tweetů v anglickém jazyce podporujících ISIS spolu s informacemi o zdrojích, ze kterých byly tweety získány. Z tohoto datasetu jsme vybrali pouze sloupec „Text“, jelikož informace z ostatních sloupců pro nás při klasifikaci textu nejsou důležité. Z textových dokumentů jsme následně odstranili odkazy na webové stránky a tagy uživatelů. Záznamy, které zůstaly po odstranění odkazů příliš krátké byly z datasetu odebrány. Zbylé záznamy jsme přidali do finálního datasetu a označili je, jako extremistické texty.

Druhým dílčím datasetem je tweet_data.csv stažený z Kaggle [64], který obsahuje přes 7000 tweetů týkajících se převážně leteckého cestování. Většina tweetů v tomto datasetu je velmi krátká a vyskytují se zde i texty v jiném než anglickém jazyce. Stejně, jako při úpravě prvního datasetu jsme text zbavili odkazů a tagů a následně jsme odstranili příliš krátké tweety. Dále jsme odstranili tweety psané v odlišných jazycích odfiltrováním takových tweetů, které neobsahovaly žádné z nejčastěji používaných anglických slov. Tím jsme samozřejmě odstranili i část anglicky psaných tweetů, ale snížení počtu záznamů bylo v tomto případě žádoucí pro zmenšení rozdílu v počtu extremistických a ne-extremistických textů. Po odstranění příliš krátkých a cizojazyčných textů obsahoval dataset cca 3000 záznamů. Nakonec jsme z něj odstranili tweety obsahující slova související s létáním, jako například: „flight“, „plane“, „ticket“. Tím jsme docílili redukce záznamů a zároveň větší rozmanitosti dat, kde je nyní poměrově větší počet náhodných tweetů, které se létání netýkají. Zbylé texty jsme následně přidali do finálního datasetu a označili je, jako ne-extremistické.

3.3.2 Implementace

Program načítá textové dokumenty s využitím knihovny pandas. Tyto dokumenty jsou následně předzpracovány pomocí funkcí z modulu preprocessing. Pro každou kombinaci klasifikačního modelu (Random forest, SVM) a metody pro extrakci příznaků (bag of words, tf-idf, Doc2Vec) vytváříme instanci třídy Component, která slouží především k uchování dosažených hodnot accuracy, precision a recall napříč iteracemi experimentu. Jelikož zatím neprovádíme ladění hyperparametrů použitých modelů, dělíme data pouze na trénovací a testovací množinu.

Extrakce příznaků, trénování modelů a vyhodnocení úspěšnosti klasifikace jednotlivých kombinací použitých metod probíhá opakovaně, přičemž data jsou pokaždé rozdělena na trénovací a testovací množinu podle jiného náhodného seedu. Výsledky těchto iterací jsou na konci průměrovány. Tento postup volíme pro dosažení odhadu přesnosti méně závislého na konkrétním rozdělení dat na trénovací a testovací množinu. Výsledky experimentu jsou zobrazovány ve formě tabulek a sloupcových grafů.

3.3.2.1 Předzpracování textu

Z textu odstraňujeme číslice a interpunkční znaménka a převádíme jej na pouze malá písmena. Při převodu textu na malá písmena může docházet ke ztrátě některých informací. Konkrétně v použitém provizorním datasetu se v extremistických textech relativně často vyskytuje zkratka „IS“, označující Islámský stát. Po převedení textu na pouze malá písmena by tato zkratka byla k nerozeznání od běžného anglického slova „is“, které je navíc stopslovem a tak by byla v dalších krocích předzpracování odstraněna. Proto v případě této zkratky učiníme výjimku a jako jedinou ji ponecháme s velkými písmeny. Kvůli takovýmto případům je důležité dataset nejdříve prostudovat a zjistit, zda se v něm nenachází podobné zdroje informací, které bychom mohli předzpracováním ztratit.

Pro tokenizaci textu využíváme v tomto případě pouze metodu split, která rozděluje text podle bílých znaků. Interpunkční znaménka byla z textu již odstraněna, a proto bychom využitím této jednoduché metody měli dostat správné rozdělení textu na jednotlivá slova. Z tokenizovaných textů dále odstraňujeme tokeny kratší, než 2 znaky, které pravděpodobně nejsou pro klasifikaci důležité.

Dalším krokem je odstranění stopslov. Využíváme seznam anglických stopslov z knihovny NLTK. V tomto seznamu se však vyskytují také slova vyjadřující zápor, jako například: „don't“, „not“, „no“. Tyto výrazy mění význam vět a jsou tedy pro klasifikaci textu důležité. Proto jsme z použitého seznamu podobná slova nejprve odstranili. Dále bylo nutné zbavit stopslova apostrofů, které se již v předzpracovaném textu nevyskytují.

Na závěr provádíme stemming tokenů pomocí třídy PorterStemmer. Ponecháváme při tom defaultní nastavení parametru určujícího stemmovací algoritmus. Tímto defaultním algoritmem je upravený Porter stemming algoritmus využívající všechna přidaná rozšíření. [46]

3.3.2.2 Extrakce příznaků

Pro extrakci příznaků z textových dokumentů byla implementována funkce extract_features. Tato funkce bere jako parametr množinu tokenizovaných trénovacích dokumentů a identifikátor metody extrakce příznaků, kterou chceme použít.

Pokud se jedná o identifikátor modelu bag-of-words, nebo tf-idf, použijeme odpovídající vectorizer a nastavíme mu parametry tak, aby se při extrakci příznaků zbytečně nespouštěl implicitní preprocesor ani tokenizer. Textové dokumenty jsou v této fázi již předzpracované. U obou modelů využíváme metodu fit_transform pro získání matice příznaků trénovacích dokumentů. Tato metoda zároveň vytváří slovník z tokenů obsažených v trénovací množině, který bude následně použit při převádění testovacích dokumentů do vektorové reprezentace.

Pokud provádíme extrakci příznaků pomocí modelu Doc2Vec, musíme nejdříve převést dokumenty do formátu TaggedDocument. Při vytváření instance třídy Doc2Vec nastavujeme následující parametry:

- **workers**: Parametr udává, kolik vláken má být použito při trénování modelu. Nastavili jsme jej tak, aby byla při výpočtu využita všechna jádra procesoru. Proces trénování modelu jinak trvá velmi dlouho jelikož v tomto experimentu trénujeme model při každé iteraci znovu na jiné trénovací množině.
- **epochs**: Parametr udává počet iterací nad daným korpusem. Čím větší je hodnota tohoto parametru, tím přesnější výsledky můžeme očekávat. Vysoký počet epoch však výrazně zpomaluje proces učení modelu. Defaultní počet iterací je 10. Hodnotu tohoto parametru jsme zvýšili na 25, jelikož pracujeme s poměrně malým datasetem, ve kterém není pro neuronovou síť snadné nalézt souvislosti mezi slovy a zachytit tak význam textu.
- **alpha**: Parametr udává počáteční rychlost učení (learning rate) neuronové sítě. Hodnota tohoto parametru se s v průběhu trénování modelu lineárně snižuje až na nastavené minimum. Defaultní hodnota tohoto parametru je nastavena na 0.025. Přesnost modelu s defaultním nastavením parametru byla velmi nízká, proto jsme hodnotu parametru alpha zvýšili na 0.05. Toto nastavení vedlo ke zvýšení úspěšnosti klasifikace modelů využívajících příznaky získané touto metodou.

U ostatních parametrů jsme ponechali defaultní hodnoty. Po vytvoření instance třídy Doc2Vec voláme metodu build_vocab, která zajistí vytvoření slovníku pro daná trénovací data. V této chvíli již můžeme model trénovat pomocí metody train. Vektory dokumentů, získané při trénování modelu jsou uložené v atributu docvecs, ze kterého tyto vektory načítáme.

Pro získání vektorové reprezentace Doc2Vec testovacích dokumentů musíme tyto dokumenty opět nejdříve převést do formátu TaggedDocument. Vektory testovacích dokumentů pak získáváme pomocí metody infer_vector třídy Doc2Vec.

3.3.2.3 Klasifikace textu

Funkci evaluate_classification předáváme jako parametr dataset rozdělený na trénovací a testovací množinu a identifikátory klasifikačního modelu a metody extrakce příznaků. Tato funkce

nejdříve provede extrakci příznaků testovacích dokumentů podle zadané metody. Získané příznaky jsou následně použity k natrénování zvoleného klasifikačního modelu. Po získání vektorové reprezentace testovacích dokumentů provádí vyhodnocení úspěšnosti klasifikace s využitím funkcí z modulu metrics. Funkce vrací nalezené hodnoty accuracy, precision a recall.

Každá iterace experimentu tedy představuje náhodné rozdělení datasetu a následné volání této funkce pro všechny kombinace klasifikačních modelů a metod extrakce příznaků. Získané hodnoty accuracy, precision a recall jsou pro každou kombinaci zvlášť ukládány a jejich průměry jsou na konci vypsány jakožto celkové výsledky experimentu.

3.3.3 Výsledky

Při experimentu bylo provedeno 50 iterací. V každé iteraci byl dataset náhodně rozdělen na trénovací a testovací množinu a pro každou kombinaci klasifikačního modelu a metody extrakce příznaků byla zjištěna úspěšnost klasifikace. Velikost testovací množiny je nastavena na 25 % z celkové velikosti datasetu. V tabulce 3.1 uvádíme průměrné hodnoty accuracy, precision a recall, které byly v průběhu iterací naměřeny pro různé kombinace použitých metod. Při opakovaném provedení experimentu se výsledky nepatrně liší, což je dáno náhodností při rozdělování datasetu v jednotlivých iteracích.

Tabulka 3.1 Tabulka výsledků prvotních experimentů

	Random forest			Support vector machine		
Použitá metoda	Accuracy	Precision	Recall	Accuracy	Precision	Recall
bag of words	0.9328	0.9936	0.6549	0.8883	1.0000	0.4207
tf-idf	0.9320	0.9950	0.6496	0.8755	1.0000	0.3524
Doc2Vec	0.9050	0.7570	0.7483	0.9221	0.9067	0.6658

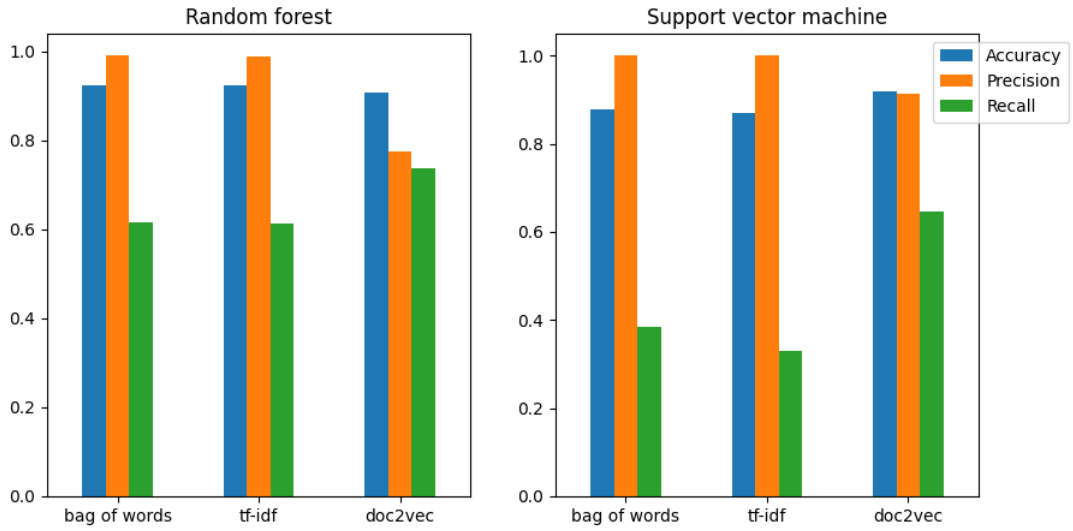
V tabulce 3.1 vidíme, že oba modely dosahují vysokých hodnot accuracy a precision. Recall je však velice nízký, zejména u modelu SVM v kombinaci s metodami tf-idf a bag-of-words. Takovéto výsledky jsou způsobeny nevyvážeností datasetu, který obsahuje pouze 190 extremistických textů a 752 ne-extremistických. Accuracy může pro nevyvážené datasety podávat zkreslené výsledky. Zde například vidíme, že klasifikační modely označily velkou část extremistických testovacích dokumentů nesprávně jako ne-extremistické. Při výpočtu accuracy však tento fakt nehraje tak velkou roli, protože extremistických dokumentů je relativně málo.

Vysoké hodnoty precision nám říkají, že testovací dokumenty, které byly při klasifikaci označeny jako extremistické, byly takto označeny správně ve většině případů. V kombinaci s nízkým recallem můžeme z těchto výsledků usoudit, že použité klasifikátory při vytváření predikcí v tomto defaultním nastavení označují jen velmi málo dokumentů jako extremistické. Toto chování klasifikačních modelů se dá ovlivnit nastavením hyperparametrů, kterým se budeme zabývat při provádění navazujících experimentů.

Dosažení vysokých hodnot accuracy je dáno zejména jednotvárností extremistických textů v použitém provizorním datasetu. Extremistické texty obsahují častá slova, podle kterých se dají od ne-extremistických textů relativně snadno odlišit.

Celkově dosáhl lepších výsledků v tomto experimentu model Random forest, a to nejen podle hodnot accuracy, ale také podle recall. Zajímavým pozorováním je, že klasifikace pomocí modelu SVM vychází nejlépe v kombinaci s modelem Doc2Vec. Random forest naopak dosahuje lepších výsledků v kombinaci s ostatními dvěma metodami extrakce příznaků. Tento výsledek potvrzuje, že je vhodné před samotnou tvorbou klasifikačního systému vyzkoušet různé kombinace metod. Žádná z použitých metod extrakce příznaků není jednoznačně lepší než ostatní. Záleží nejen na kombinaci s klasifikačním modelem, ale také na způsobu předzpracování textu, použitém datasetu a dalších faktorech.

Obrázek 3.1 Výsledky prvotních experimentů



3.4 Trénovací program

Trénovací program vytváří a ukládá klasifikátory, které jsou následně načítány výsledným systémem pro detekci extremistických textů. Každý klasifikátor je tvořen kombinací natrénovaných modelů pro extrakci příznaků, redukci dimenzionality a klasifikaci.

Program provádí experimenty na komplexnějším datasetu obsahujícím české texty. Na základě výsledků těchto experimentů vybíráme několik klasifikátorů, které jsou tímto programem uloženy. Výsledky experimentů zároveň určují váhu uložených modelů, která je zohledněna při jejich hlasování ve výsledném systému.

3.4.1 Popis vytvořených datasetů

V rámci této práce byly vytvořeny 3 různé datasety českých textů: extremistický, neutrální a dataset s ne-extremistickými texty zabývajícími se extremismem. Datasety byly vytvářeny manuálním výběrem vhodných textů z různých zdrojů tak, aby byla zaručena relevance a zároveň určitá rozmanitost záznamů. Datasety jsou uloženy ve formátu CSV, a každý z nich obsahuje kromě samotných textů také sloupec s odkazy na jejich zdroje (pokud není dostupný odkaz, obsahuje tento sloupec název publikace), a sloupec s označením typu zdroje, například „bakalářská práce“ či „analytická zpráva“. Délka textů se pohybuje v rozmezí od jedné do osmi vět.

Extremistický dataset
Tento dataset byl vytvořen ve spolupráci s vedoucí práce, která poskytla seznam zdrojů vhodných pro sběr potřebných dat a zároveň dodala část extremistických textů, které jsou v tomto datasetu obsaženy. Jako hlavní zdroj textů pro tento dataset byl použit český překlad knihy White Power [65] napsaná zakladatelem Americké nacistické strany, Georgem Lincolnem Rockwellem. Úryvky z této knihy tvoří téměř polovinu záznamů všech záznamů. Velkou část datasetu tvoří také překlady textů z White Supremacist datasetu dostupného na úložišti Zenodo [63]. Většinu ostatních zdrojů tvoří odborné publikace citující neonacistické výroky.

Dataset obsahuje celkem 350 textů, které se dají označit za neonacistické či nacistické. Při detekci extremismu se tedy omezujeme pouze na tyto vybrané ideologie. Použité texty se vyznačují

zejména následujícími vlastnostmi:

- Nenávist vůči Židům, přistěhovalcům či lidem různých ras
- Propagace nadřazenosti bílé (árijské) rasy
- Vyzývání k rasovému násilí či válce
- Obdiv k Adolfu Hitlerovi
- Popírání holokaustu

Neutrální dataset
Obsahuje celkem 400 textů, které nejsou extremistické a nijak s extremismem nesouvisí. Při sběru dat byl kladen důraz na různorodost záznamů, které byly sbírány z různých zdrojů, jako například: komentáře v diskuzích pod zpravodajskými články, recenze na filmy, úryvky článků, příspěvky v diskusních fórech a další.

Dataset o extremismu
Texty obsažené v extremistickém a neutrálním datasetu by měly být relativně snadno odlišitelné na základě rozdílné slovní zásoby. Při trénování modelu pouze na těchto datech bychom mohli očekávat detekování extremismu pro libovolný textu obsahující například slova: „rasa“, „Hitler“, „Žid“, která se v neutrálních textech nevyskytují. Pokoušíme se vytvořit model, schopný odlišit například popis historických událostí spojených s nacismem od extremistických výroků obhajujících holokaust. Při trénování takového modelu je zapotřebí využít také ne-extremistické texty vztahující se k problematice neonacismu.

Proto byl vytvořen tento dataset obsahující 200 záznamů. Jedná se zejména o úryvky z publikací zaměřených na popis ideologie neonacismu. Texty zahrnují například: popis extremistických organizací, popis historických událostí v nacistickém Německu, definice souvisejících pojmů, rozbor základních myšlenek neonacismu a další. V tomto datasetu jsou dále zahrnuty texty, které přímo s extremismem nesouvisí, ale obsahují slova používaná v extremistických textech v jiném kontextu. Texty obsažené v tomto datasetu jsou označeny, jako ne-extremistické stejně, jako texty z neutrálního datasetu.

3.4.2 Implementace trénovacího programu

Program načítá data z připravených českých datasetů. Z každého načteného datasetu je odebráno 15 procent záznamů podle pevně daného seedu, tato část dat je určena pro vyhodnocení úspěšnosti výsledného klasifikačního systému. Trénovací program s těmito daty nepracuje, pouze je ukládá samostatně do CSV souborů. V opačném případě by docházelo k ovlivnění odhadu přesnosti výsledného klasifikačního systému.

Program po spuštění zobrazuje následující nabídku operací:

- Vyhodnocení experimentu
- Nalezení nejlepších hyperparametrů Doc2Vec
- Nalezení nejlepších hyperparametrů klasifikačních modelů
- Uložení natrénovaných modelů

Abychom získali uložené klasifikátory pro výsledný systém, provádíme nejdříve ladění hyperparametrů. Hledáme vhodné hodnoty parametrů pro každou kombinaci klasifikačního modelu a modelu pro extrakci příznaků zvlášť. Hodnoty příznaků získané tímto laděním jsou pro každou kombinaci modelů ukládány do souboru. Při vytváření instancí daných modelů využíváme tyto připravené parametry. Ladění hyperparametrů je časově náročné, a proto jsme tento krok oddělili od zbytku implementace.

Následuje provedení operace „Vyhodnocení experimentu“. V tomto kroku vycházíme z implementace prvotních experimentů, pracujeme však s komplexnějším českým datasetem a zkoumáme také různé metody předzpracování českých textů a využití n-gramů. Po provedení této operace program zobrazí tabulky s výsledky. Podle těchto výsledků nastavujeme, které kombinace metod (klasifikátory) mají být uloženy. Operace „Uložení natrénovaných modelů“ pouze provede trénování každého z vybraných klasifikátorů a uloží jej do souboru. Testovací množinu dat, která je v trénovacím programu použita pro vyhodnocení experimentů můžeme chápat, jako validační množinu výsledného systému. Pomáhá tento systém optimalizovat pomocí výběru vhodných dílčích klasifikátorů.

Popis implementace trénovacího programu dále upřesníme v následujících částech, které zahrnují informace o struktuře program, postup při ladění hyperparametrů, implementaci a vyhodnocení provedených experimentů a ukládání vybraných klasifikátorů.

3.4.2.1 Struktura programu

Trénovací program je členěn do následujících modulů:

- **main**: Je hlavní modul obsahující především rozhraní pro načítání dat z uložených datasetů a komunikaci s uživatelem prostřednictvím příkazové řádky. Na základě zadaného vstupu určuje která operace má být provedena.
- **preprocessing**: Obsahuje funkce pro předzpracování textu.
- **feature_ extraction**: Obsahuje funkce, které provádí extrakci příznaků podle zvoleného modelu. Dále je zde implementováno ladění hyperparametrů modelu Doc2Vec.
- **experimetns**: Obsahuje funkce a třídy využívané při provádění experimentů.
- **display_results**: Obsahuje funkce, které vypisují na standartní výstup přehledné tabulky s výsledky experimentů.
- **czech_stemmer**: Upravený modul pro stemming českých slov převzatý z [59].
- **classification**: Obsahuje funkce pro trénování klasifikačních modelů, vyhodnocení přesnosti klasifikace, redukci dimenzionality a ladění hyperparametrů klasifikačních modelů.

3.4.2.2 Ladění hyperparametrů

Při ladění hyperparametrů hledáme vhodné hodnoty hyperparametrů pro oba klasifikační modely Random forest, SVM a také pro model Doc2Vec. Používáme předzpracované textové dokumenty ze všech tří datasetů. Výsledné hodnoty parametrů jsou uloženy do souborů ve formátu PKL a zároveň vypsány na standardní výstup.

Při ladění hyperparametů klasifikačních modelů využíváme 4-fold cross validaci pomocí třídy GridSearchCV. Pro přiřazení skóre různým kombinacím hodnot hyperparametrů je nastavena metrika balanced accuracy, která představuje průměrný recall při klasifikaci extremistických a ne-extremistických textů. Hyperparametry jsou laděny a ukládány zvlášť pro každou kombinaci klasifikačního modelu a metody extrakce příznaků.

Při ladění hyperparametrů modelu Doc2Vec dělíme data na trénovací a validační množinu. Hyperparametry ladíme zvlášť pro každý klasifikační model. Ze zadaných hodnot parametrů jsou vytvořeny všechny možné kombinace s využitím třídy ParameterGrid. Pro každou kombinaci pak provádíme inicializaci třídy Doc2Vec s danými hodnotami hyperparametrů, kterou použijeme k extrakci příznaků testovacích dokumentů. Na těchto příznacích natrénujeme klasifikační model a zjistíme accuracy tohoto modelu na validační množině. Kombinace hodnot hyperparametrů modelu Doc2Vec, se kterou bylo dosaženo nejvyšší accuracy je pak vybrána jako nejlepší pro daný klasifikační model.

Níže jsou uvedeny konkrétní hyperparametry a jejich hodnoty, které byly při ladění použity.

Random forest

- **min_samples_split**: Parametr určuje minimální počet vzorků, které se musí nacházet v uzlu stromu, aby se tento uzel mohl dále dělit. Defaultně je hodnota tohoto parametru nastavena na 1. Pro omezení komplexnosti dílčích stromů tuto hodnotu zvyšujeme. Tím dosáhneme vyšší robustnosti modelu Random forest a předcházíme přeučení modelu. Při ladění jsou nastaveny hodnoty: 35, 40 a 45.
- **criterion**: Parametr definuje metriku pro určení nejlepšího splitu při trénování dílčích stromů. Defaultně je touto metrikou Gini index. Při ladění parametr nastavujeme na Gini index a Entropii (tedy na hodnoty „gini“ a „entropy“).
- **n_estimators**: Určuje počet stromů tvořících model Random forest. Parametr při ladění nastavujeme na: 150 a 200.

Ladění provádíme zvlášť pro každou metodu extrakce příznaků zkoumanou v experimentech. Těchto metod je celkem 7 a budou dále popsány v sekci „Experimenty“ 3.4.2.3. Získáváme tedy 7 různých kombinací hodnot hyperparametrů, které vychází nejlépe pro danou vektorovou reprezentaci textových dokumentů.

V šesti z těchto sedmi případů byl laděním nastaven počet dílčích stromů na 200. Hodnoty zbylých dvou parametrů se pro různé vektorové reprezentace velmi liší.

SVM

- **class_weight**: Parametr určuje váhu, dat různých kategorií. Defaultně je nastaven na None. To znamená, že všechny kategorie mají stejnou váhu. Při ladění jsme využili toto defaultní nastavení a zároveň mód „balanced“, který nastavuje kategoriím váhy nepřímo úměrné jejich četnosti v datech. Data z méně zastoupené kategorie tedy mají vyšší váhu.
- **probability**: Určuje, zda mají být při trénování využity odhady pravděpodobností. Pokud je tento parametr nastaven na True, pravděpodobnosti jsou s využitím cross validace kalibrovány pomocí Plattova škálování logistické regrese na skóre modelu SVM. Plattovo škálování je způsob transformace výstupu modelu na pravděpodobnostní rozdělení kategorií. Při ladění nastavujeme parametr na True a False.
- **decision_function_shape**: Parametr definuje použitou rozhodovací funkci. Může nabývat dvou hodnot: „ovo“ (one-vs-one) či „ovr“ (one-vs-rest). Defaultně je parametr nastaven na „ovr“. Při ladění zkoušíme obě uvedené možnosti.
- **kernel**: Specifikuje typ kernelové funkce použité v algoritmu. Defaultně je nastaven na „rbf“ (radial basis function). Při ladění používáme defaultní funkci „rbf“, a také lineární a sigmoidní kernelové funkce.
- **gamma**: Parametr určuje hodnotu gamma koeficientu pro sigmoidní, polynomiální a RBF kernel. Pro tento parametr jsou předdefinované dvě hodnoty: „auto“ a „scalle“, které při ladění nastavujeme.

Stejně, jako u modelu Random forest provádíme ladění hyperparametrů tohoto modelu pro každou metodu extrakce příznaků zvlášť. U některých příznaků byla ve všech případech nastavena stejná hodnota. Toto se týká příznaku „probability“, který byl laděním nastaven pokaždé na True a příznaku „decision_function_shape“, pro který vychází lépe funkce one-vs-one. Parametr „class_weight“ byl ve většině případů nastaven do módu „balanced“. Ostatní parametry vychází pro různé vektorové reprezentace textů různě.

Doc2Vec

- **dm**: Parametr určuje algoritmus použitý při trénování modelu. Nabývá dvou hodnot: 1 pro algoritmus Distributed memory a 0 pro DBOW. Obě možnosti byly při ladění použity. Při defaultním nastavení modelu je parametr nastaven na 1.

- **epochs**: Parametr byl již podrobněji popsán v sekci „Prvotní experimenty“ 3.3.2.2. Udává počet iterací nad daným korpusem. Při ladění mu byly nastaveny hodnoty: 25 a 30.
- **min_count**: Parametr určuje minimální celkový počet výskytů slov. Slova, která mají v trénovacích dokumentech nižší počet výskytů, než je tato stanovená hranice, jsou při vytváření vektorů ignorována. Tomuto parametru byly nastaveny hodnoty: 2, 3 a 4.
- **window**: Parametr definuje maximální vzdálenost mezi predikovaným slovem a slovy použitými pro jeho predikování. Určuje tedy velikost kontextu použitého při trénování modelu. Parametr je defaultně nastaven na hodnotu 5. Při ladění mu byly nastaveny hodnoty: 4, 5 a 6
- **vector_size**: Určuje velikost vektorů reprezentujících textové dokumenty. Defaultní hodnota parametru je 100. Toto hodnotu jsme se rozhodli zvýšit vzhledem k tomu, že použitý dataset obsahuje i delší texty, pro které by takto malé vektory nemuseli být dostačující. Při ladění byly parametru nastaveny hodnoty: 200 a 250.
- **alpha**: Parametr byl již popsán v sekci „Prvotní experimenty“ 3.3.2.2. Udává počáteční learning rate neuronové sítě. Při ladění mu byly nastaveny hodnoty: 0.055, 0.06, 0.065 a 0.07.

Hyperparametry získané laděním jsou pro oba klasifikační modely velmi podobné. Jako nejlepší byly v obou případech vybrány hodnoty: „window“ = 6, „min_count“ = 3, „epochs“ = 25 a „alpha“ = 0.065. Lišila se pouze hodnota parametru „vector_size“, který byl při klasifikací pomocí modelu Random forest nastaven na 200, zatímco pro model SVM byl nastaven na 250.

3.4.2.3 Návrh experimentů

Před samotným spuštěním experimentu určujeme, zda chceme pracovat pouze s extremistickým a neutrálním datasetem, nebo také s datasetem o extremismu. Experimenty, prováděné trénovacím programem vychází z implementace výše popsaných prvotních experimentů. Stojí tedy na principu opakovaného vyhodnocování úspěšnosti pro všechny kombinace zkoumaných metod a následného průměrování těchto výsledků pro každou z kombinací. Tato implementace je podrobněji popsána v sekci 3.3.2. Zde uvádíme pouze ty části programu, které se v prvotních experimentech nevyskytují.

Experimenty prováděné trénovacím programem pracují s datasety českých textů. Proto nyní volíme odlišný postup při předzpracování textových dokumentů. Pro převedení českých slov do základního tvaru využíváme dva různé nástroje: modul czech_stemmer pro stemming a knihovnu Simplemma pro lemmatizaci slov. Oba nástroje jsou popsány v sekci „Použité nástroje“ 3.2. Při experimentech zkoumáme úspěšnost klasifikace nejen v závislosti na použité metodě extrakce příznaků a klasifikačním modelu, ale porovnáváme také tyto dva nástroje pro předzpracování textu.

Implementace se liší také využitím n-gramů při extrakci příznaků. V experimentech zjišťujeme, jaký vliv má na úspěšnost klasifikace právě využití n-gramů různé délky. Při extrakci příznaků pomocí modelů bag-of-words a tf-idf využíváme následující n-gramů:

- Pouze 1-gramy
- 1 až 2-gramy
- 1 až 3-gramy

Zároveň nastavujeme těmto modelům parametr „max_features“ na 4000. Tato hodnota byla vybrána na základě pozorování, že počet příznaků při využití pouze 1-gramů se pohybuje kolem této hodnoty. Tento parametr omezuje počet příznaků, vytvářených při využití více různých n-gramů najednou. Použity jsou pouze příznaky s nejvyššími vahami. Extrakce příznaků pomocí modelu Doc2Vec probíhá stejně, jako v prvotních experimentech. Rozdíl je pouze v tom, že používáme při inicializaci modelu parametry získané laděním.

Dále provádíme redukci dimenzionality založenou na selekci relevantních příznaků. Pomocí nástroje SelectKBest vybíráme 80 % příznaků, které dosahují nejvyššího skóre, stanoveného na základě analýzy rozptylu ANOVA.

3.4.2.4 Vyhodnocení experimentů

Výsledky tvoří průměrné hodnoty accuracy, precision a recall z 50 iterací experimentu. Stejně, jako při provádění prvotních experimentů, jsou data rozdělována na trénovací a testovací množinu podle náhodného seedu. Výsledky se tedy mohou při opakovaném provedení stejného experimentu nepatrně lišit. Testovací množinu tvoří 25 % záznamů.

V této sekci vyhodnocujeme výsledky experimentů při použití dvou různých datasetů. Jako první uvádíme výsledky experimentu na datasetu složeném z extremistických a neutrálních textů. Při výběru vhodných klasifikátorů pro finální klasifikační systém pro nás budou směrodatné výsledky druhého experimentu, který využívá texty ze všech tří českých datasetů.

Neutrální a extremistické texty

V následujících tabulkách jsou uvedeny výsledky experimentu v závislosti na použité metodě předzpracování textu. Klasifikační modely byly v tomto případě trénovány pouze na textech z neutrálního a extremistického datasetu.

Tabulka 3.2 Výsledky při použití lemmatizace

	Random forest			Support vector machine		
Extrakce příznaků	Accuracy	Precision	Recall	Accuracy	Precision	Recall
bag of words 1-gramy	0.8680	0.9274	0.7731	0.8556	0.8915	0.7809
bag of words 1 až 2-gramy	0.8660	0.9340	0.7617	0.8539	0.8970	0.7701
bag of words 1 až 3-gramy	0.8661	0.9337	0.7625	0.8559	0.9019	0.7698
tf-idf 1-gramy	0.8606	0.9237	0.7600	0.8824	0.8942	0.8447
tf-idf 1 až 2-gramy	0.8602	0.9222	0.7601	0.8811	0.8905	0.8460
tf-idf 1 až 3-gramy	0.8599	0.9212	0.7602	0.8819	0.8910	0.8471
doc2vec	0.8336	0.8278	0.8068	0.8206	0.8534	0.7402

Tabulka 3.3 Výsledky při použití stemmingu

	Random forest			Support vector machine		
Extrakce příznaků	Accuracy	Precision	Recall	Accuracy	Precision	Recall
bag of words 1-gramy	0.8766	0.9318	0.7954	0.8640	0.8869	0.8153
bag of words 1 až 2-gramy	0.8758	0.9382	0.7872	0.8626	0.8893	0.8089
bag of words 1 až 3-gramy	0.8756	0.9373	0.7874	0.8626	0.8903	0.8072
tf-idf 1-gramy	0.8696	0.9241	0.7862	0.8845	0.9019	0.8468
tf-idf 1 až 2-gramy	0.8698	0.9291	0.7823	0.8838	0.9001	0.8466
tf-idf 1 až 3-gramy	0.8685	0.9236	0.7844	0.8842	0.8997	0.8485
doc2vec	0.8533	0.8412	0.8474	0.8480	0.8536	0.8158

Na základě výsledků uvedených v tabulkách 3.2 a 3.3 můžeme usoudit, že při použití stemmingu dosahují oba klasifikační modely vyšší přesnosti než při lemmatizaci textu. Dále z těchto výsledků vyplývá, že použití n-gramů v tomto případě nepřináší zvýšení přesnosti klasifikace. Celkově nejvyšší accuracy v tomto experimentu dosahuje model SVM v kombinaci s extrakcí příznaků metodou tf-idf a předzpracováním textu pomocí stemmingu.

Texty ze všech datasetů

V následujících tabulkách jsou opět uvedeny výsledky experimentu v závislosti na použité metodě

předzpracování textu.

Tabulka 3.4 Výsledky při použití lemmatizace

	Random forest			Support vector machine		
Extrakce příznaků	Accuracy	Precision	Recall	Accuracy	Precision	Recall
bag of words 1-gramy	0.8149	0.8186	0.6386	0.8148	0.7669	0.7135
bag of words 1 až 2-gramy	0.8177	0.8431	0.6204	0.8130	0.7706	0.7009
bag of words 1 až 3-gramy	0.8157	0.8403	0.6173	0.8143	0.7802	0.6898
tf-idf 1-gramy	0.8155	0.8180	0.6424	0.8426	0.7904	0.7786
tf-idf 1 až 2-gramy	0.8177	0.8250	0.6417	0.8410	0.7850	0.7821
tf-idf 1 až 3-gramy	0.8173	0.8192	0.6465	0.8413	0.7860	0.7815
doc2vec	0.7766	0.6852	0.7279	0.7566	0.6542	0.7211

Tabulka 3.5 Výsledky při použití stemmingu

	Random forest			Support vector machine		
Extrakce příznaků	Accuracy	Precision	Recall	Accuracy	Precision	Recall
bag of words 1-gramy	0.8290	0.8405	0.6679	0.8247	0.7832	0.7296
bag of words 1 až 2-gramy	0.8280	0.8587	0.6443	0.8235	0.7845	0.7229
bag of words 1 až 3-gramy	0.8290	0.8600	0.6474	0.8259	0.7930	0.7194
tf-idf 1-gramy	0.8306	0.8388	0.6733	0.8482	0.8018	0.7869
tf-idf 1 až 2-gramy	0.8293	0.8445	0.6637	0.8493	0.8032	0.7879
tf-idf 1 až 3-gramy	0.8317	0.8491	0.6650	0.8470	0.7997	0.7855
doc2vec	0.8076	0.7530	0.7209	0.7796	0.7023	0.7147

Charakter výsledků je velmi podobný, jako v předchozím experimentu s využitím pouze neutrálních a extremistických textů. V tabulkách 3.4 a 3.5 vidíme, že vyšší přesnost klasifikace vychází při předzpracování pomocí stemmingu. Celkově nejvyšší accuracy dosáhl opět model SVM s využitím stemmingu pro předzpracování a s metodou tf-idf využívající 1-gramy a 2-gramy.

Můžeme si dále všimnout, že došlo k celkovému zhoršení přesnosti oproti předchozímu experimentu. Tento výsledek byl očekávaný vzhledem k tomu, že zde používáme také texty, které se nedají od těch extremistických snadno odlišit na základě pouhých výskytů slov souvisejících s danou ideologií. Pro správnou klasifikaci těchto textů je třeba uvažovat kontext použití daných slov. Proto byly očekávány nejlepší výsledky při použití modelu Doc2Vec, který by měl zachycovat ve vektorové podobě celkový význam textových dokumentů. Tento model však podle uvedených výsledků dosahuje nejnižší přesnosti v porovnání s ostatními metodami extrakce příznaků. Příčinou může být nedostatečná velikost datasetu pro naučení tohoto modelu či neoptimální výběr hodnot hyperparametrů při jejich ladění.

Model SVM zpravidla dosahuje vyšších hodnot recall než Random forest. Při klasifikaci pomocí modelu Random forest naopak vychází celkově vyšší precision. Na základě výsledků tohoto experimentu vybíráme celkem 6 různých kombinací metod, které budou dohromady tvořit dílčí klasifikátory výsledného systému. Pro výsledný systém volíme následující kombinace metod:

- Stemming + všechny varianty metody tf-idf + SVM
- Lemmatizace + tf-idf s 1-gramy a 1-gramy až 3-gramy + SVM
- Stemming + bag-of-words s 1-gramy až 3-gramy + Random forest

Poslední kombinace využívající model Random forest je zde zahrnuta pro zvýšení rozmanitosti dílčích klasifikátorů. Ostatní kombinace jsou vybrány na základě vysokých hodnot accuracy a recall v tomto experimentu.

3.4.2.5 Ukládání modelů

Kombinace metod, vybrané na základě výsledků experimentů jsou uloženy ve slovníku s názvem „best_models“. Pokud je vybrána operace „Uložení natrénovaných modelů“, provádíme nejprve předzpracování textů ze všech tří českých datasetů pomocí odpovídající metody. Pro každou kombinaci ze slovníku pak trénujeme odpovídající modely pro extrakci příznaků a klasifikaci, spolu s modelem pro redukci dimenzionality. Tyto natrénované modely pak ukládáme do souborů pomocí funkce drop z Joblib. Každý uložený klasifikátor má svou vlastní složku, ve které jsou uloženy tyto modely ve formátu PKL. Do této složky ukládáme také textový soubor obsahující hodnotu accuracy, které daná kombinace metod dosáhla v experimentu se všemi texty. Složky s uloženými modely pak stačí manuálně přesunout do adresáře s výsledným klasifikačním systémem.

3.5 Systém pro detekci extremistických textů

Systém je navržen, jako konzolová aplikace implementovaná v jazyce Python. Zkráceně jej označujeme pouze jako klasifikační systém. Vzhledem k tomu, že extremistický dataset použitý pro trénování klasifikátorů obsahuje zejména neonacistické a nacistické texty, dokáže systém rozpoznávat pouze tyto ideologické směry. Provádí detekci extremistických textů na základě váženého hlasování dílčích klasifikátorů, které byly vytvořeny trénovacím programem. Váhy jsou určeny hodnotou accuracy, které daný klasifikátor dosáhl v experimentech provedených trénovacím programem. Tyto klasifikátory jsou tvořeny před-trénovanými modely pro extrakci příznaků, redukci dimenzionality a klasifikaci. V následující tabulce je uveden přehled dílčích klasifikátorů.

Tabulka 3.6 Přehled dílčích klasifikátorů a jejich vah

Klasifikační model	Způsob předzpracování	Metoda extrakce příznaků	Váha
SVM	stemming	tf-idf s 1-gramy	0.8482
SVM	stemming	tf-idf s 1-gramy a 2-gramy	0.8493
SVM	stemming	tf-idf s 1-gramy až 3-gramy	0.8470
SVM	lemmatizace	tf-idf s 1-gramy	0.8426
SVM	lemmatizace	tf-idf s 1-gramy až 3-gramy	0.8413
Random forest	stemming	bag-of-words s 1-gramy až 3-gramy	0.8317

Systém umožňuje zvolit jeden ze tří módů zadáním argumentu při spuštění. Při spuštěni s argumentem -h jsou vypsány uživatelské instrukce. Jednotlivé módy umožňují provádět následující operace:

- Detekce extremismu pro text zadaný na standardním vstupu. Systém vypisuje na standardní výstup predikci, zda se jedná o extremistický text či nikoli.
- Hromadné načtení a klasifikace textů ze souboru. Na standardní výstup, jsou vypsány jednotlivé texty označené jako „extremist“ či „not extremist“. Soubor musí být ve správném formátu, který je specifikován v přiloženém souboru README.txt.
- Odhad přesnosti klasifikace textu s využitím testovacích dat. Výstupem jsou dosažené hodnoty accuracy, precision, recall a f1-score.

3.5.1 Implementace

Funkce main nejprve provádí načtení zadaných argumentů a uložených klasifikátorů. Na základě načtených argumentů pak volí příslušný mód. Pokud není zadán žádný argument, je provedena detekce extremismu pro text zadaný na standardním vstupu. Uživatel je vyzván k zadání textu

do příkazové řádky. Získaný text je následně předán funkci get_partial_predictions, která pro každý dílčí klasifikátor provádí předzpracování textu podle příslušné metody, extrakci příznaků, redukci dimenzionality a následnou klasifikaci pomocí natrénovaných modelů daného klasifikátoru. Tato funkce vrací pole dvojic, přičemž každá dvojice obsahuje predikovanou kategorii a váhu klasifikátoru, který tuto predikci učinil. Tento výstup je předán funkci compute_final_prediction, která provádí vážené hlasování. Při hlasování jsou váhy klasifikátorů, které určily text, jako neextremistický převedeny na zápornou hodnotu. Pokud je součet všech takto upravených vah kladný, text je označen jako extremistický.

Pokud je zvolen mód pro hromadné načtení a klasifikaci textů ze souboru, provádíme postupné načítání textu po řádcích. Pro každou načtenou řádku predikujeme, zda se jedná o extremistický soubor či nikoli stejným způsobem, jako v předchozím případě. Na standardní výstup postupně vypisujeme texty označené predikovanou kategorií. Načítání souboru končí při nalezení první prázdné řádky obsahující pouze odřádkování.

Pokud je zvolen mód pro odhad přesnosti klasifikace systému, dotážeme se uživatele, zda chce při tomto odhadu využít pouze extremistické a neutrální texty, nebo texty ze všech testovacích datasetů. Podle získané odpovědi načteme příslušné testovací datasety a předáme je funkci classify_test_documents. Tato funkce nejdříve provádí předzpracování textu. Poté nechá každý dílčí klasifikátor vytvořit predikce pro testovací data. Pro každý textový dokument je pak volána funkce compute_final_predictions a její výstupy jsou ukládány. S využitím funkcí z modulu metrics vyhodnotíme accuracy, precision, recall a f1-skóre finálních predikcí a tyto hodnoty vypíšeme na standardní výstup.

3.5.2 Odhad přesnosti

Pro odhad přesnosti systému využíváme připravené testovací datasety, přičemž extremistický dataset obsahuje 53 záznamů, neutrální dataset obsahuje 61 záznamů a dataset o extremismu obsahuje 30 záznamů. Hodnoty v následující tabulce jsou výstupem systému v módu pro odhad přesnosti.

Tabulka 3.7 Odhad přesnosti výsledného systému

Metrika	Extremistikcé a neutrální texty	Všechny texty
Accuracy	0.868421	0.854166
Precision	0.931818	0.82
Recall	0.773584	0.773584
f1-score	0.845360	0.796116

Na základě hodnot uvedených v tabulce 3.7 je odhadovaná přesnost klasifikačního systému přibližně 85 %. Zároveň na základě těchto výsledků odhadujeme, že systém dokáže detekovat přibližně 77 % extremistických textů. V tabulce je vidět porovnání úspěšnosti při klasifikaci pouze extremistických a neutrálních textů s úspěšností při klasifikaci textů ze všech tří datasetů. Rozdíl mezi nimi již není zdaleka tak vysoký, jako ve výsledcích experimentů trénovacího programu 3.4.2.4, což je způsobeno tím, že pro trénování dílčích klasifikátorů byly využity texty ze všech datasetů.

Systém pro detekci extremistických textů má vyšší precision než recall. To znamená, že určitou část extremistických textů není schopen detekovat. Pokud však označí text jako extremistický, je zde relativně vysoká šance, že je tato predikce správná.

3.6 Diskuze

Vytvořený klasifikační systém dosahuje relativně vysoké přesnosti vzhledem k omezené velikosti použitých datasetů. Při trénování modelů pro klasifikaci textu je důležité použít dostatečný

vzorek dat. Pro tento účel se často používají datasety obsahující tisíce či deseti tisíce záznamů. Pro vytvoření takto velkého datasetu však nebyly nalezeny dostatečně obsáhlé zdroje extremistických textů v češtině.

Trénovací program umožňuje relativně snadné přetrénování klasifikátorů na jiných datasetech. Je zde tedy možnost pro zvýšení robustnosti klasifikačního systému při využití větších a komplexnějších datasetů pro trénování dílčích klasifikátorů. Zároveň by bylo možné tímto způsobem rozšířit okruh extremistických ideologií, které je systém schopen detekovat. V současné době je omezen pouze na rozpoznávání neonacistických či nacistických textů.

Klasifikační systém by se dal rozšířit také využitím většího počtu klasifikačních modelů strojového učení. V této práci jsou pro klasifikaci textu využity modely Random forest a SVM. Přidáním dalších modelů, například Logistické regrese či Naivního Bayesova klasifikátoru bychom docílili vyšší rozmanitosti dílčích klasifikátorů ve výsledném klasifikačním systému a tedy vyšší robustnosti.

Kapitola 4
Závěr

Cílem této práce bylo navrhnout a implementovat vhodný klasifikátor pro detekci extremistických textů. V rámci této práce byly vytvořeny dva různé programy: trénovací program a klasifikační systém. Trénovací program slouží k vytváření a ukládání klasifikátorů, které jsou následně využity v klasifikačním systému. Výběr vhodných metod pro vytvoření těchto klasifikátorů je založen na provedených experimentech. Konkrétně byly pro klasifikaci textu použity modely SVM a Random forest. Klasifikační systém pouze načítá dílčí klasifikátory ze souborů a vytváří predikce na základě jejich váženého hlasování.

Vytvořený klasifikační systém pracuje s českými texty a je zaměřen na detekci neonacistických či nacistických textů. Systém byl implementován, jako konzolová aplikace v jazyce Python. Umožňuje klasifikaci jednotlivých textů zadaných na standardním vstupu či hromadnou klasifikaci textů ze zadaného souboru. Při odhadu přesnosti klasifikace na testovacích datech dosáhl klasifikační systém accuracy 85 %. Cíl práce byl tedy splněn.

V rámci této práce bylo provedeno několik experimentů. Při těchto experimentech byla vyhodnocována úspěšnost klasifikace v závislosti na použité metodě předzpracování textu, metodě pro extrakci příznaků a také na použitém klasifikačním modelu. Zároveň byla porovnávána přesnost klasifikace při využití pouze textů z extremistického a neutrálního datasetu s přesností při využití textů ze všech tří datasetů. Na základě výsledků těchto experimentů byly vybrány vhodné kombinace metod pro dílčí klasifikátory výsledného klasifikačního systému.

Cílem rešeršní části práce bylo seznámení čtenáře s metodami klasifikace textu a vytvoření přehledu související literatury týkající se detekce extremistických textů. V této části práce byly nejprve vymezeny pojmy týkající se extremismu. Dále byl popsán vývoj metod používaných pro detekci extremismu v online prostoru. Byly zde představeny konkrétní publikace zabývající se detekcí extremismu z různých časových období. Rešeršní část práce dále obsahuje popis konkrétních metod a postupů používaných při klasifikaci textu s využitím modelů supervizovaného strojového učení. Tématem detekce extremistických textů se zabývá velké množství publikací. Na konci rešeršní části práce uvádíme několik z nich spolu s popisem konkrétních metod, které byly pro detekci extremistických textů využity. Cíl rešeršní části práce byl tedy splněn, rešerše zahrnuje důkladný popis využívaných metod pro klasifikaci textu a přehled související literatury.

Dalším dílčím cílem této práce bylo vytvoření datasetu obsahujícího extremistické i neextremistické texty. V rámci této práce byly vytvořeny tři různé datasety českých textů. Prvním z nich je extremistický dataset, který obsahuje celkem 350 neonacistických či nacistických textů. Texty byly vybírány z různých zdrojů, například z neonacistických knih či publikací citujících neonacistické výroky. Dále byl vytvořen neutrální dataset obsahující 400 ne-extremistických textů, které s extremismem nijak nesouvisí. Třetí dataset obsahuje ne-extremistické texty, které se však vztahují k vybraným extremistickým ideologiím. Jedná se například o popisy historických událostí souvisejících s nacismem.

Bibliografie

1. BERGER, J.M. Extremist Construction of Identity: How Escalating Demands for Legitimacy Shape and Define In-Group and Out-Group Dynamics. In: The International Centre for Counter-Terrorism, 2017. ISSN 2468-0656. Dostupné z DOI: 10.19165/2017.1.07.
2. STRIEGHER, Jason-Leigh. Violent-extremism: An examination of a definitional dilemma. In: SRI Security Research Institute, Edith Cowan University, Perth, Western, 2015.
3. *Co je extremismus - Ministerstvo vnitra České republiky* [online]. 2010 [cit. 2022-04-01]. Dostupné z: https://www.mvcr.cz/clanek/co-je-extremismus.aspx.
4. HUCKIN, Thomas. Propaganda defined. *Propaganda and rhetoric in democracy: History, theory, analysis*. 2016, s. 125–127. ISBN 9780809335077.
5. MCGOWAN, Lee. *Radikální pravice v Německu: Od roku 1870 po současnost*. 2004. ISBN 80-7260-122-9.
6. CHARVÁT, Jan. *Současný politický extremismus a radikalismus*. 2007. ISBN 978-80-7367-098-6.
7. AGARWAL, Swati; SUREKA, Ashish. Applying social media intelligence for predicting and identifying on-line radicalization and civil unrest oriented threats. *arXiv preprint arXiv:1511.06858*. 2015.
8. MALIK, Nikita. Terror in the Dark: How Terrorists use Encryption, the Darknet, and Cryptocurrencies. 2018. ISBN 978-1-909035-45-4.
9. RAY, Beverly; MARSH, George E. Recruitment by extremist groups on the Internet. 2001.
10. GAIKWAD, Mayur; AHIRRAO, Swati; PHANSALKAR, Shraddha; KOTECHA, Ketan. Online extremism detection: A systematic literature review with emphasis on datasets, classification techniques, validation methods, and tools. *IEEE Access*. 2021, roč. 9, s. 48364–48404.
11. ZHOU, Yilu; REID, Edna; QIN, Jialun; CHEN, Hsinchun; LAI, Guanpi. US domestic extremist groups on the Web: link and content analysis. *IEEE intelligent systems*. 2005, roč. 20, č. 5, s. 44–51.
12. FU, Tianjun; HUANG, Chun-Neng; CHEN, Hsinchun. Identification of extremist videos in online video sharing sites. In: *2009 IEEE International Conference on Intelligence and Security Informatics*. 2009, s. 179–181.
13. SCANLON, Jacob R; GERBER, Matthew S. Automatic detection of cyber-recruitment by violent extremists. *Security Informatics*. 2014, roč. 3, č. 1, s. 1–10.
14. JOHNSTON, Andrew H.; WEISS, Gary M. Identifying sunni extremist propaganda with deep learning. In: *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. 2017, s. 1–6. Dostupné z DOI: 10.1109/SSCI.2017.8280944.

15. LINDHOLM, Andreas; WAHLSTRÖM, Niklas; LINDSTEN, Fredrik; SCHÖN, Thomas B. Supervised machine learning. *Department of Information Technology, Uppsala University: Uppsala, Sweden*. 2019, s. 112.

16. *BI-VZD přednáška 1 handout 29. září 2021* [online]. 2021 [cit. 2022-04-14]. Dostupné z: `https://courses.fit.cvut.cz/BI-VZD/lectures/files/BI-VZD-01-cs-handout.pdf`.

17. GASPARETTO, Andrea; MARCUZZO, Matteo; ZANGARI, Alessandro; ALBARELLI, Andrea. A Survey on Text Classification Algorithms: From Text to Predictions. *Information*. 2022, roč. 13, č. 2.

18. KOWSARI, Kamran; JAFARI MEIMANDI, Kiana; HEIDARYSAFA, Mojtaba; MENDU, Sanjana; BARNES, Laura; BROWN, Donald. Text classification algorithms: A survey. *Information*. 2019, roč. 10, č. 4.

19. LOURDUSAMY, Ravi; ABRAHAM, Stanislaus. A survey on text pre-processing techniques and tools. *International Journal of Computer Sciences and Engineering*. 2018, roč. 6, č. 03.

20. LEUSCH, Gregor; UEFFING, Nicola; VILAR, David; NEY, Hermann. Preprocessing and normalization for automatic evaluation of machine translation. In: *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*. 2005, s. 17–24.

21. SARICA, Serhad; LUO, Jianxi. Stopwords in technical language processing. *Plos one*. 2021, roč. 16, č. 8.

22. JIVANI, Anjali Ganesh et al. A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*. 2011, roč. 2, č. 6.

23. NUŢU, Maria. Deep Learning Approach for Automatic Romanian Lemmatization. *Procedia Computer Science*. 2021, roč. 192, s. 49–58.

24. *BI-VZD přednáška 12 handout 21. dubna 2021* [online]. 2021 [cit. 2022-04-01]. Dostupné z: `https://kam.fit.cvut.cz/bi-vzd/lectures/files/BI-VZD-12-cs-handout.pdf`.

25. MIKOLOV, Tomas; CHEN, Kai; CORRADO, Greg; DEAN, Jeffrey. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*. 2013.

26. LE, Quoc; MIKOLOV, Tomas. Distributed Representations of Sentences and Documents. In: *Proceedings of the 31st International Conference on Machine Learning* [online]. Bejing, China: PMLR, 2014, sv. 32, s. 1188–1196 [cit. 2022-05-06]. Proceedings of Machine Learning Research, č. 2. Dostupné z: `https://proceedings.mlr.press/v32/le14.html`.

27. LAU, Jey Han; BALDWIN, Timothy. An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv preprint arXiv:1607.05368*. 2016.

28. DENG, Xuelian; LI, Yuqing; WENG, Jian; ZHANG, Jilian. Feature Selection for Text Classification: A Review. *Multimedia Tools Appl.* 2019, roč. 78, č. 3. ISSN 1380-7501. Dostupné z DOI: `10.1007/s11042-018-6083-5`.

29. IKONOMAKIS, Emmanouil; KOTSIANTIS, Sotiris; TAMPAKAS, V. Text Classification Using Machine Learning Techniques. *WSEAS transactions on computers*. 2005, roč. 4. ISBN 1109-2750.

30. VERNER, Pavel. *Metody výběru příznaků pro klasifikační a regresní úlohy*. 2017. Dipl. pr. České vysoké učení technické v Praze.

31. *BI-VZD přednáška 10 handout 25. dubna 2022* [online]. 2022 [cit. 2022-05-02]. Dostupné z: `https://courses.fit.cvut.cz/BI-VZD/lectures/files/BI-VZD-10-cs-handout.pdf`.

32. *BI-VZD přednáška 5 handout 14. března 2022* [online]. 2022 [cit. 2022-04-01]. Dostupné z: `https://courses.fit.cvut.cz/BI-VZD/lectures/files/BI-VZD-04-cs-handout.pdf`.

33. REFAEILZADEH, Payam; TANG, Lei; LIU, Huan. Cross-Validation. In: *Encyclopedia of Database Systems*. Ed. LIU, LING; ÖZSU, M. TAMER. Boston, MA: Springer US, 2009, s. 532–538. ISBN 978-0-387-39940-9. Dostupné z DOI: 10.1007/978-0-387-39940-9_565.
34. *K nearest neighbor: Knn algorithm: KNN in Python amp; R* [online]. 2020 [cit. 2022-04-14]. Dostupné z: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/.
35. *BI-VZD přednáška 6 handout 9. listopadu 2021* [online]. 2021 [cit. 2022-04-01]. Dostupné z: https://courses.fit.cvut.cz/BI-VZD/lectures/files/BI-VZD-06-cs-handout.pdf.
36. *BI-VZD přednáška 9 handout 4. dubna 2022* [online]. 2022 [cit. 2022-04-14]. Dostupné z: https://courses.fit.cvut.cz/BI-VZD/lectures/files/BI-VZD-09-cs-handout.pdf.
37. MAMMONE, Alessia; TURCHI, Marco; CRISTIANINI, Nello. Support vector machines. *Wiley Interdisciplinary Reviews: Computational Statistics*. 2009, roč. 1, č. 3, s. 283–289.
38. PRADHAN, Ashis. Support vector machine-a survey. *International Journal of Emerging Technology and Advanced Engineering*. 2012, roč. 2, č. 8, s. 82–85.
39. KULKARNI, Vrushali Y; SINHA, Pradeep K. Pruning of Random Forest classifiers: A survey and future directions. In: *2012 International Conference on Data Science Engineering (ICDSE)*. 2012, s. 64–68. Dostupné z DOI: 10.1109/ICDSE.2012.6282329.
40. *BI-VZD lecture 3 handout 28. února 2022* [online]. 2022 [cit. 2022-04-14]. Dostupné z: https://courses.fit.cvut.cz/BI-VZD/lectures/files/BI-VZD-02-cs-handout.pdf.
41. MUSSIRALIYEVA, Shynar; BOLATBEK, Milana; OMAROV, Batyrkhan; MEDETBEK, Zhanar; BAISPAY, Gulshat; OSPANOV, Ruslan. On Detecting Online Radicalization and Extremism Using Natural Language Processing. In: *2020 21st International Arab Conference on Information Technology (ACIT)*. 2020, s. 1–5. Dostupné z DOI: 10.1109/ACIT50332.2020.9300086.
42. SHARIF, Waqas; MUMTAZ, Shahzad; SHAFIQ, Zubair; RIAZ, Omer aj. An Empirical Approach for Extreme Behavior Identification through Tweets Using Machine Learning. *Applied Sciences*. 2019, roč. 9, č. 18. ISSN 2076-3417. Dostupné z DOI: 10.3390/app9183723.
43. ASHCROFT, Michael; FISHER, Ali; KAATI, Lisa; OMER, Enghin; PRUCHA, Nico. Detecting Jihadist Messages on Twitter. In: *2015 European Intelligence and Security Informatics Conference*. 2015, s. 161–164. Dostupné z DOI: 10.1109/EISIC.2015.27.
44. *Natural Language Toolkit* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://www.nltk.org.
45. PORTER, Martin F. An algorithm for suffix stripping. *Program*. 1980.
46. *Source code for nltk.stem.porter* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://www.nltk.org/_modules/nltk/stem/porter.html.
47. *Source code for nltk.tokenize* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://www.nltk.org/_modules/nltk/tokenize.html.
48. PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL V., aj. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 2011, roč. 12, s. 2825–2830.
49. *Sklearn.ensemble.randomforestclassifier* [online]. 2022 [cit. 2022-05-06]. Dostupné z: scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.
50. *Sklearn.svm.SVC* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

51. *Sklearn.feature_extraction.text.CountVectorizer* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html?highlight=countvectorizer#sklearn.feature_extraction.text.CountVectorizer.
52. *Sklearn.feature_extraction.text.TfidfVectorizer* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html?highlight=tfidfvectorizer#sklearn.feature_extraction.text.TfidfVectorizer.
53. *Sklearn.model_selection.GRIDSEARCHCV* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html?highlight=gridsearchcv#sklearn.model_selection.GridSearchCV.
54. *Sklearn.model_selection.train_test_split* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html?highlight=train_test_split#sklearn.model_selection.train_test_split.
55. *Sklearn.feature_selection.Selectkbest* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html?highlight=selectkbest#sklearn.feature_selection.SelectKBest.
56. *API reference* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://scikit-learn.org/stable/modules/classes.html?highlight=metrics#module-sklearn.metrics.
57. ŘEHŮŘEK, Radim. *Gensim: Topic modelling for humans* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://radimrehurek.com/gensim/intro.html.
58. *Simplemma* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://pypi.org/project/simplemma/.
59. GOMES, Luís. *Czech_stemmer* [online]. 2010 [cit. 2022-05-06]. Dostupné z: https://research.variancia.com/czech_stemmer/.
60. DOLAMIC, Ljiljana; SAVOY, Jacques. Indexing and stemming approaches for the Czech language. *Information Processing Management*. 2009, roč. 45, č. 6, s. 714–720. ISSN 0306-4573. Dostupné z DOI: https://doi.org/10.1016/j.ipm.2009.06.001.
61. *Persistence* [online]. 2021 [cit. 2022-05-06]. Dostupné z: https://joblib.readthedocs.io/en/latest/persistence.html#persistence.
62. *Pandas documentation* [online]. 2022 [cit. 2022-05-06]. Dostupné z: https://pandas.pydata.org/docs/.
63. GAIKWAD, Mayur; AHIRRAO, Swati; PHANSALKAR, Shraddha; KOTECHA, Ketan. *Multi-ideology ISIS/Jihadist White Supremacist (MIWS) Dataset for Multi-class Extremism Text Classification* [online]. Zenodo, 2021. Ver. V1 [cit. 2022-04-14]. Dostupné z DOI: 10.5281/zenodo.5687447.
64. CHENG, Jacob. *Sample Twitter Data for Text Analytics* [online]. Kaggle, 2020. Version 1 [cit. 2022-04-14]. Dostupné z: https://www.kaggle.com/datasets/chengjhj/tweet-data.
65. ROCKWELL, George Lincoln. *White Power* [online]. 1966 [cit. 2022-05-06]. Dostupné z: https://docplayer.cz/8196465-White-power-prekladatel.html.

Obsah přiloženého média

- `README.txt` ..... stručný popis obsahu média
- `src`
  - `classification_system` ..... zdrojové kódy klasifikačního systému
  - `training_program` ..... zdrojové kódy trénovacího programu
  - `starting_experiments` ..... zdrojové kódy prvotních experimentů
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text`
  - `thesis.pdf` ..... text práce ve formátu PDF