Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra kybernetiky

DIPLOMOVÁ PRÁCE

Klasifikace dokumentů s částečnou informací od učitele

Srpen 2021 Bc. Ondřej MACEK

Abstrakt

Tato práce se zabývá klasifikací dokumentů s částečnou informací od učitele. Teoretická část zastřešuje základy strojového učení a definuje prostor pro metody učení s částečnou informací od učitele mezi metodami učení s učitelem a bez učitele. Dále jsou popsány metody učení s částečnou informací od učitele a klasifikace dokumentů jako teoretický základ pro vývoj vlastní metody. Následuje popis vyvinutého algoritmu pro zvětšení objemu trénovacích dat pomocí neanotované datové sady, doplněný výsledky experimentů.

Klíčová slova

Strojové učení, učení s částečnou informací od učitele, klasifikátor, klasifikace dokumentů, Bag-of-words, Python, scikit-learn

Abstract

Macek, Ondřej, Bc. *Semi-supervised document classification* [*Klasifikace dokumentů s částečnou informací od učitele*]. Pilsen, 2021. Diploma thesis (in Czech). University of West Bohemia. Faculty of Applied Sciences. Department of Cybernetics. Supervisor: Doc. Ing. Pavel Ircing, Ph.D.

The thesis deals with the semi-supervised document classification. The theoretical part covers the basics of machine learning and defines the space for semi-supervised learning methods between supervised and unsupervised learning. Furthermore, sections of semi-supervised learning methods and document classification cover the theoretical basis for developing the own method. The own developed algorithm for enlargement the training data using an unannotated dataset is explained, accompanied by experimental results.

Key words

Machine learning, semi-supervised learning, classifier, document classification, Bag-of-words, Python, scikit-learn

Prohlášení

Předkládám tímto k posouzení a obhajobě diplomovou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne 25. srpna 2021 .....

Podpis

Poděkování

Tímto bych chtěl poděkovat panu Doc. Ing. Pavlu Ircingovi, Ph.D. za skvělé vedení diplomové práce, poskytování cenných rad, za čas věnovaný konzultacím a trpělivost při vypracování této práce.

Dále bych rád poděkoval mé rodině a přátelům, zejména pak Ing. Martinu Káši, Ing. Šárce Klímkové, Bc. Martině Mackové a Pavle Ronovské kteří mě po dobu mého studia a při psaní této práce podporovali, zásobovali radami a přispívali k větší motivaci.

Obsah

1

Úvod

Odjakživa se člověk snažil postavit robota, který by za něj vykonával práci. Takovým robotem může být i software, který zpracovává nebo třídí data. Takových systémů a programů je již spousta a usnadňují člověku např. práci s e-mailovou komunikací (spamové filtry), se získáváním informací (vyhledávače) nebo jako virtuální asistenti, kteří dokáží porozumět mluvené řeči a zpětně uživateli odpovědět.

S příchodem digitalizace začal člověk sbírat a ukládat nepřeberné množství dat, které můžou být velmi cenné, jestliže z nich dokáže vytáhnout správnou informaci. Dnes je na světě dat již tolik, že se jimi ručně žádný člověk neprobere, a další přibývají. Taková data můžou využít např. firmy ke svému růstu, ať už z hlediska zkvalitnění služeb pro zákazníky, snížení provozních nákladů nebo objevení potenciální díry na trhu. Problémem je, že většina těchto dat je nestrukturalizovaná a není dostatečně nebo vhodně anotována pro použití ve standardních metodách strojového učení, jako je učení s učitelem, které pro svojí práci potřebují dostatečné množství anotovaných dat, které dnes zpravidla nejsou k dispozici.

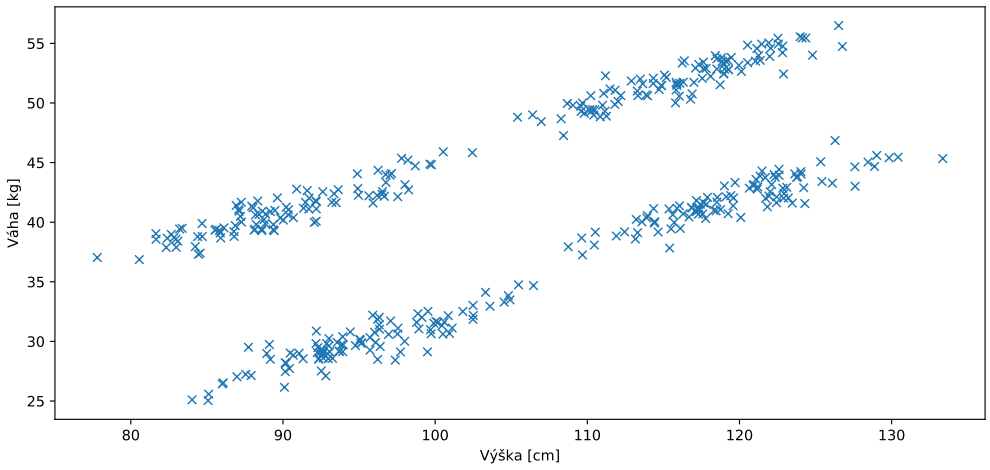
Metody s částečnou informací od učitele můžou být v mnoha případech dobrou volbou pro práci s velkým množstvím neanotovaných dat, protože se v datech dokáží samy „zorientovat“ na základě malého množství ručně anotovaných dat, které je člověk schopný ručně připravit během několika hodin nebo jednotek dní. Klasifikace dokumentů s částečnou informací od učitele, kterou se zabývá tato práce, tak může být jednou z úloh, která může pomoci dnes nezpracovatelné množství dat využít.

2

Strojové učení

Před představením učení s částečnou informací od učitele v kapitole 4, které je hlavním tématem této práce, je vhodné popsat základy strojového učení, ze kterých obecně metody učení klasifikátorů vycházejí.

Pro představení bude použit následující jednoduchý příklad, který ilustruje, jak lze z určité situace vytvořit úlohu strojového učení. Kdyby na Zemi přistáli mimozemšťané, bylo by tak možné přivítat např. 100 malých zelených mužíků, které předtím nikdo neviděl. Ti by měli, podobně, jako lidé, popsatelné a měřitelné rozdíly, použitelné pro klasifikaci. Prvních parametrů kterých by si někdo mohl všimnout může být např. jejich váha a výška. Po provedeném měření je pak na obr. 2.1 vidět, jak by mohl vypadat graf 100 mužíků, podle jejich váhy a výšky.



Obr. 2.1: Zobrazení rozdělení obrazů zelených mužíků podle měření váhy a výšky.

S využitím těchto dat je už možné řešit různé úlohy. Je možné mužíky třídit podle jejich váhy nebo výšky, dělat z nich skupiny, najít mezi nimi extrémní případy, zkusit predikovat

jednu hodnotu na základě druhé, atd. Následující části definují za pomoci uvedeného příkladu základy strojového učení [Zhu, Goldberg, 2009].

2.1 Data

Všechny učící se algoritmy jsou závislé na datech, která k učení používají. Tedy můžou se naučit rozpoznávat situace, které data obsahují, ale ne ty, které v datech obsaženy nejsou. Objekt nebo také obraz $\mathbf{x}$ reprezentuje konkrétní instanci, která je definovaná jako D-dimenzionální příznakový vektor, popisující její vlastnosti $\mathbf{x} = (x_1, ..., x_D) \in \mathbb{R}^D$. Každá dimenze tak popisuje jednu vlastnost objektu.

Reprezentace objektu pomocí příznakového vektoru nese všechny jeho informace, které jsou o daném objektu k dispozici. Například dva malí zelení mužíčci se stejnou váhou a výškou jsou pomocí váho-výškové reprezentace nerozeznatelní, i když se třeba každý jinak jmenuje. Pro matematický popis označujeme tučným $\mathbf{x}$ celou instanci objektu a x_d d-tý příznak $\mathbf{x}$. V uvedeném příkladu se příznakový vektor skládá z příznaků: váha x_1 a výška x_2. Příznaky také mohou obsahovat diskrétní hodnoty, např. příznak 0 = nemá tykadla, 1 = má tykadla. V případě několika obrazů, se značí x_{id}, jako d-tý příznak i-té instance.

Data jsou často rozdělována do podmnožin, např. trénovací, testovací nebo validační, nad kterými se provádí různé úlohy. Trénovací množinou se myslí kolekce obrazů $\{\mathbf{x}_i\}_{i=1}^n = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, která je použita jako vstupní množina učícího procesu. V příkladu se zelenými mužíky je trénovací množina složena z $n = 100$ obrazů $\mathbf{x}_1, \ldots, \mathbf{x}_{100}$. Je to v zásadě „zkušenost“, která je dána učícímu se algoritmu, ze které čerpá. Co se z ní může naučit je různé a záleží na samotném algoritmu. Dva základní koncepty, které využívají trénovací množinu, jsou ve strojovém učení: učení bez učitele a učení s učitelem.

2.2 Učení bez učitele

Algoritmus učení bez učitele používá trénovací množinu n obrazů $\{\mathbf{x}_i\}_{i=1}^n$. Součástí algoritmu není žádný učitel, který by dohlížel na správné zařazení obrazů nebo poskytoval jakoukoliv další informaci, než je zanesena v datech trénovací množiny. Mezi nejznámější úlohy učení bez učitele patří:

- shlukování - rozdělení obrazů do skupin
- detekce anomálií - identifikace malého množství případů, které se liší od většiny

- snížení dimenze - převedení obrazů do nové reprezentace s nižší dimenzí příznakového vektoru

Nejrozšířenější úlohou učení bez učitele je bezpochybně shlukování. To rozdělí množinu $\{\mathbf{x}_i\}_{i=1}^n$ do k shluků tak, aby si byly obrazy ve stejném shluku podobné a obrazy v různých shlucích byly rozdílné. Počet shluků k může zadat uživatel nebo může být odvozen ze samotného trénovacího vzorku. Jsou shlukovací algoritmy, které potřebují jako vstupní parametr, počet shluků, do kterých data rozdělují, další se řídí podle jiných parametrů a počet shluků znát dopředu nepotřebují.

Kolik shluků je možné najít v datech malých zelených mužíčků na obr. 2.1? Možná $k = 2$, $k = 4$ nebo více. Bez dalších informací a předpokladů jsou přijatelné obě varianty. Na rozdíl od učení s učitelem (představené v následující části), neexistuje nikdo, kdo by mohl říct, které případy by měly být v jednotlivých shlucích.

2.3 Anotovaná a neanotovaná data

Dá se předpokládat, že mimozemšťané z příkladu mají pohlaví: ženské nebo mužské (takže by všichni, přece jenom, neměli být nazýváni zelenými *mužíčky*). Když se ke každému obrazu přidá informace o jeho zařazení do třídy, neboli jeho označení (angl. label), je možné zjistit, zda-li lze předpovědět pohlaví konkrétního mimozemšťana, na základě jeho váhy a výšky. Případně s jiným označením pomocí váhy a výšky zjistit, jestli se jedná o mladistvého nebo dospělého jedince.

Jinými slovy, značka y je požadovaná předpověď pro obraz $\mathbf{x}$. Takové značky mohou být z konečné množiny hodnot, např. {ženy, muži}. Každá taková různá hodnota je nazývána třídou. Třídy jsou obvykle označovány celými čísly, např. žena = -1, muž = 1, tedy $y \in \{-1, 1\}$. Toto konkrétní označení se často používá pro binární (dvoutřídní) rozdělení a takovéto třídy se pak obecně nazývají negativní a pozitivní třída. Pro úlohy s více než dvěma třídami se tradičně používá označení $y \in \{1, \ldots, C\}$, kde C je počet tříd. Označení tříd obecně nemá žádnou spojitost s jejich umístěním v prostoru. To znamená, že dvě třídy, označené $y = 1$ a $y = 2$ nemusí být nutně vzájemně blíž, než dvě třídy $y = 1$ a $y = 3$. Značky mohou nabývat také spojitých hodnot v množině $\mathbb{R}$. Například se lze pokusit předpovědět krevní tlak malého zeleného mimozemšťana na základě jeho váhy a výšky.

Při učení s učitelem se trénovací množina skládá z datových párů, z nichž každý obsahuje obraz $\mathbf{x}$ a značku třídy y: $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$. Kde y je značka pro obraz $\mathbf{x}$, který poskytuje expert, neboli učitel. Odtud název učení s učitelem. Takové datové páry (obraz,

značka) se nazývají *anotovaná data*, zatímco samotné obrazy bez označení (jako při učení bez učitele) se nazývají *neanotovaná data*.

2.4 Učení s učitelem

Nechť existuje množina obrazů $\mathcal{X}$ a množina jejich značek $\mathcal{Y}$. Nechť je $P(\mathbf{x}, y)$ (neznámé) sdružené rozdělení pravděpodobnosti obrazů a značek $\mathcal{X} \times \mathcal{Y}$. Pro daný trénovací vzorek, kde každá třída je nezávislá a má stejné rozdělení pravděpodobnosti (angl. independent and identically distributed, neboli i.i.d.) $\{(x_i, y_i)\}_{i=1}^{n} \overset{\text{i.i.d.}}{\sim} P(\mathbf{x}, y)$ se při učení s učitelem trénuje funkce $f : \mathcal{X} \mapsto \mathcal{Y}$ v nějaké rodině funkcí $\mathcal{F}$ s cílem, aby $f(\mathbf{x})$ předpovídala pravdivé zařazení y také na budoucích datech $\mathbf{x}$, kde $(\mathbf{x}, y) \overset{\text{i.i.d.}}{\sim} P(\mathbf{x}, y)$.

Podle toho, jestli je množina $\mathcal{Y}$ diskrétní nebo spojitá se učení s učitelem dále rozděluje na *klasifikaci* a *regresi*. Klasifikace je proces učení funkce f, která se nazývá *klasifikátor* nad diskrétní množinou $\mathcal{Y}$. Práce se spojitou $\mathcal{Y}$ je tedy regrese a f je pak nazývána *regresní funkce*.

Jak se pozná dobrá funkce f? Podle definice je nejlepší f

$$f = \underset{f \in \mathcal{F}}{argmin}\, \mathbb{E}_{\mathbf{x},y\sim P}[c(\mathbf{x}, y, f(\mathbf{x}))], \tag{2.1}$$

kde $argmin$ znamená „nalezení f, které minimalizuje následné kritérium“. $\mathbb{E}_{\mathbf{x},y\sim P}[\cdot]$ je střední hodnota chyby predikce (angl. expectation) klasifikátoru nad náhodnými testovacími daty vybranými z pravděpodobnostního rozdělení P. $c(\cdot)$ je ztrátová funkce, která určuje cenu nebo dopady chybné klasifikace. Je důležité, že je pozornost omezena jen na určitou rodinu funkcí $\mathcal{F}$, a to především z výpočetních důvodů. Pokud by toto omezení bylo odstraněno a v úvahách byly všechny možné funkce, výsledná f^* by byla Bayesovým optimálním prediktorem, tedy to nejlepší, co lze v průměru očekávat. Pro rozdělení P bude tato funkce při predikci nést nejmenší možnou ztrátu. Kritérium $\mathbb{E}_{\mathbf{x},y\sim P}[c(\mathbf{x}, y, f^*(\mathbf{x}))]$ je známé jako *Bayesova chyba*. Bayesův optimální prediktor však nemusí být obecně omezen na rodinu funkcí $\mathcal{F}$. Cílem tedy je najít f, které se co nejvíce blíží Bayesově optimálnímu prediktoru, ale v rodině funkcí $\mathcal{F}$ se nachází.

Je důležité zmínit, že základní rozdělení $P(\mathbf{x}, y)$ je neznámé, a proto není možné f^* přímo najít nebo jinak měřit výkonnost jakéhokoliv prediktoru f. V tom spočívá základní potíž statistického strojového učení. Je třeba zobecnit predikci z konečného trénovacího vzorku na jakákoli neviděná testovací data. Tento přístup je známý jako *indukce*.

Pro posouzení výkonnosti f je možné využít aproximaci založenou na chybě trénovacího vzorku. To znamená, že neznámou střední hodnotu chyby nahradíme průměrem

z trénovacího vzorku. Pro trénovací vzorek $\{(x_i, y_i)\}_{i=1}^{n}$ je chyba trénovacího vzorku rovna $\frac{1}{n}\sum_{i=1}^{n} c(\mathbf{x}_i, y_i, f(\mathbf{x}_i))$. Pro klasifikaci se nejčastěji používá ztrátová funkce se ztrátou 0-1 $c(\mathbf{x}, y, f(\mathbf{x})) \equiv f(\mathbf{x}_i) \neq y_i$:

$$\frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i) \neq y_i, \tag{2.2}$$

kde $f(\mathbf{x}) = 1$, když f predikuje pro x jinou třídu, než y, $f(\mathbf{x}) = 0$ jinak. Jednou z nejčastěji používaných ztrátových funkcí pro regresi je s druhou mocnina ztráty $c(\mathbf{x}, y, f(\mathbf{x})) \equiv (f(\mathbf{x}_i) - y_i)^2$:

$$\frac{1}{n}\sum_{i=1}^{n}(f(\mathbf{x}_i) - y_i)^2. \tag{2.3}$$

Je nutné dávat pozor na lákavou představu minimalizování chyby trénovacího vzorku. Tato strategie je však chybná, protože takové f pak bude mít tendenci nadměrně vyhovovat konkrétnímu trénovacímu vzorku a nastane tzv. *přetrénování* klasifikátoru (angl. overfitting). To znamená, že statistický šum, který je v konkrétním trénovacím vzorku, ovlivní klasifikátor natolik, že se bude zdát trénování sice úspěšnější, ale samotný klasifikátor se bude učit jiný vztah, než ten skutečný mezi $\mathcal{X}$ a $\mathcal{Y}$. Takto přetrénovaný prediktor bude mít malou chybu trénovacího vzorku, ale bude mít pravděpodobně horší výsledky na budoucích testovacích datech. Problematika přetrénování je další dílčí oblastí v rámci strojového učení, která se nazývá teorie počítačového učení. Zavádí přísné vazby mezi chybou trénovacího vzorku a skutečnou chybou pomocí formálního pojetí složitosti, jako je Vapnik-Chervonenkisova dimenze nebo Rademacherova složitost. Na základě informací z teorie počítačového učení je jedním z rozumných způsobů trénování strategie je hledat f, které „téměř“ minimalizuje chybu trénovacího vzorku, a zároveň regulovat f tak, aby nebylo v určitém smyslu příliš složité.

K odhadu budoucí výkonnosti f lze použít samostatný vzorek označených případů, který se nazývá *testovací vzorek*: $\{(x_j, y_j)\}_{j=n+1}^{n+m} \overset{\text{i.i.d.}}{\sim} P(\mathbf{x}, y)$. Testovací vzorek se při trénování nepoužívá, a proto poskytuje věrný (nezkreslený) odhad budoucí výkonnosti. Odpovídající vztah pro chybu testovacího vzorku pro klasifikaci se ztrátou 0-1 je

$$\frac{1}{m}\sum_{j=n+1}^{n+m} f(\mathbf{x}_j) \neq y_j \tag{2.4}$$

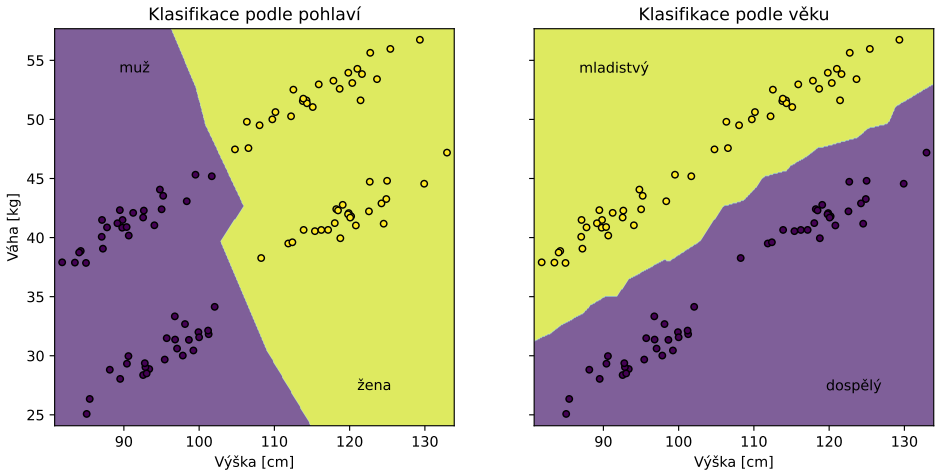
a pro regresi s druhou mocninou ztráty

$$\frac{1}{m}\sum_{j=n+1}^{n+m} (f(\mathbf{x}_j) - y_j)^2. \tag{2.5}$$

Na D-rozměrný příznakový vektor obrazu $\mathbf{x}^*$ lze pohlížet jako na bod v D-rozměrném prostoru. Klasifikátor přiřadí každému bodu v prostoru příznaků značku, čímž se prostor

příznaků rozdělí na rozhodovací oblasti, v jejichž rámci jsou body stejně označkovány. Hranice oddělující tyto oblasti se nazývá *rozhodovací rovina* (popř. nadrovina pro vyšší dimenze D) navržená klasifikátorem.

Navázáním na dříve uvedený příklad je možné natrénovat klasifikátor na určení pohlaví nebo věku pomocí váhy a výšky malých zelených mimozemšťanů. Např. použitím klasifikace podle k-nejbližších sousedů při $k = 1$ je možné zobrazit rozdělující roviny jako na grafu na obr. 2.2.



Obr. 2.2: Možnosti klasifikace 100 mimozemšťanů se zobrazením rozdělující roviny.

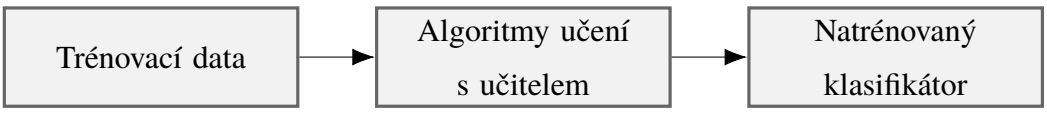
3

Klasifikátory

Klasifikátor je z hlediska strojového učení funkce f, která klasifikuje diskrétní množinu obrazů $\mathcal{Y}$ do předem známých tříd $\mathcal{C}$. Postupů neboli algoritmů, které definují, jakým způsobem se klasifikační funkce adaptuje na trénovací sadu dat, resp. učí, je mnoho a zpravidla se název *klasifikátor* používá pro označení celého klasifikačního algoritmu, který zmíněnou funkci ladí. Klasifikační funkce i klasifikační algoritmus si jsou dostatečně blízké na to, aby bylo možné používat pojem klasifikátor pro oba z nich.

Jestliže je cílem strojového učení navrhnout takový klasifikátor, který dokáže klasifikovat dosud neviděné obrazy, je nutné mít k dispozici nejprve data, na kterých se takový klasifikátor naučí, aby následně klasifikoval zmíněné dosud neviděné obrazy. To přináší dvě fáze klasifikace, které jsou znázorněny na následujících diagramech.

Fáze učení



Testovací fáze

V první fázi, kterou je učení, klasifikátory navrhují na základě dostupných trénovacích dat matematický model, tedy funkci f. Časová náročnost algoritmu je závislá na velikosti korpusu. Čím větší korpus je, tím větší je časová náročnost. Na druhou stranu, čím více dat má klasifikátor k dispozici, tím přesnější může být výsledná klasifikační funkce. Druhou fází je testovací a/nebo aplikační fáze. V této fázi se modely vytvořené učícími se algoritmy používají k předpovídání výstupů y_i z dat, které učící se algoritmy ve fázi učení neviděly.

Před vlastní aplikací se vždy provádí validační fáze, která slouží k ověření přesnosti modelů vyvinutých v první fázi [Huang a kol., 2006].

V následující části jsou popsány klasifikátory přinášející různé přístupy k učení a následné klasifikaci, které jsou dále použity ve vlastní práci. Kromě výsledné klasifikace vrací zmíněné klasifikátory hodnotu skóre spolehlivosti. Ta je buď určena přímo pravděpodobností zařazení, která je výhodná, protože již sama nabývá ze své podstaty hodnot od 0 do 1, nebo hodnotou vzdálenosti obrazu od rozhodovací funkce. Druhá zmíněná nabývá kladných hodnot pro třídu, do které by měl být obraz zařazen, záporných hodnot pro ostatní třídy. Může nastat situace, kdy jsou hodnoty obrazu pro všechny třídy záporné. V takovém případě se obraz nachází na „druhé“ straně rozdělovací nadroviny. Nicméně stále platí, že nejvyšší hodnota ukazuje na nejpravděpodobnější zařazení. Jestliže je žádoucí používat skóre spolehlivosti obou variant, je vhodné je nejprve normalizovat.

3.1 Multinomiální naivní Bayesův klasifikátor

Rodina klasifikátorů *Naive Bayes* je souborem algoritmů učení s učitelem založených na použití Bayesovy věty s „naivním“ předpokladem podmíněné nezávislosti každé dvojice příznaků dané třídy. Bayesova věta uvádí následující vztah při dané třídě y a závislém vektoru příznaků od x_1 do x_n:

$$P(y|x_1,\ldots,x_n) = \frac{P(y)P(x_1,\ldots,x_n|y)}{P(x_1,\ldots,x_n)}. \tag{3.1}$$

Při použití naivního předpokladu podmíněné nezávislosti platí, že

$$P(x_i|y,x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n) = P(x_i|y), \tag{3.2}$$

což lze zjednodušit pro všechny i na:

$$P(y|x_1,\ldots,x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1,\ldots,x_n)}. \tag{3.3}$$

Protože $P(x_1,\ldots,x_n)$ je pro všechny třídy konstantní, je možné vztah zjednodušit na relativní četnost třídy v trénovací množině (apriorní pravděpodobnost třídy) $P(y)$ a součin aposteriorní pravděpodobnosti $\prod_{i=1}^{n} P(x_i|y)$:

$$P(y|x_1,\ldots,x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y), \tag{3.4}$$

z čehož vyplývá, že pro odhad zařazení obrazu $\mathbf{x} = \{x_1,\ldots,x_n\}$ je možné vybrat takovou třídu, pro kterou je apriorní pravděpodobnosti třídy a součin aposteriorní pravděpodobnosti maximální:

$$\hat{y} = \arg\max_y P(y)\prod_{i=1}^{n} P(x_i|y). \tag{3.5}$$

Různé naivní Bayesovy klasifikátory se liší především předpoklady, které přijímají ohledně rozdělení parametrů $P(x_i|y)$.

Navzdory svým zdánlivě příliš zjednodušeným předpokladům fungují naivní Bayesovy klasifikátory v porovnání se složitějšími algoritmy poměrně dobře. Proto se používají v mnoha reálných situacích, jako např. pro klasifikaci dokumentů nebo filtrování spamu. K odhadu potřebných parametrů potřebují jen malé množství trénovacích dat.

Díky tomu, že je algoritmicky možné oddělit jednotlivé příznaky podmíněného rozdělení třídy, umí být naivní Bayesovy klasifikátory ve srovnání se sofistikovanějšími metodami extrémně rychlé. Tímto oddělením může být každé rozdělení odhadováno nezávisle jako jednorozměrné rozdělení, což velmi usnadňuje práci s vysokodimenzionálními daty.

Multinomiální naivní Bayesův klasifikátor

Jedná se o implementaci základního naivního Bayesova algoritmu pro vícerozměrně rozložená data, která se často používají při klasifikaci textu, kde jsou data obvykle reprezentována jako počty slov vektorů nebo vektory TF-IDF. Rozdělení je pro každou třídu y parametrizováno vektory $\theta_y = (\theta_{y1}, \ldots, \theta_{yn})$, kde n je počet příznaků, neboli velikost slovníku pro klasifikaci textu, a θ_{yi} je pravděpodobnost $P(x_i|y)$ příznaku vyskytujícího se v obrazu patřícím do třídy y.

Parametry θ_y se odhadují pomocí vyhlazené verze maximální věrohodnosti, tj. počítáním relativních četností:

$$\hat{\theta_{yi}} = \frac{N_{yi} + \alpha}{N_y + \alpha n}, \tag{3.6}$$

kde $N_{yi} = \sum_{x \in T} x_i$ je počet výskytů příznaku ve vzorku třídy y v trénovací množině T a N_y je celkový počet všech příznaků třídy y.

Vyhlazovací koeficienty zohledňují příznaky, které se nevyskytují v trénovací množině, a zabraňují nulovým pravděpodobnostem v nadcházejících výpočtech. Pro parametr $\alpha = 1$ se vztah 3.6 nazývá Laplaceovo vyhlazování, zatímco pokud je $\alpha < 1$, tak se vztahu říká Lidstoneovo vyhlazování [Pedregosa a kol., 2011].

3.2 Klasifikace pomocí k-nejbližších sousedů

Principem metod, které ke klasifikaci využívají obrazy z okolí je najít předem definovaný počet trénovacích obrazů, které jsou nejblíže aktuálně zkoumanému. Na základě jeho k-nejbližších sousedů (angl. k-nearest neighbors, kNN) je předpovězeno označení zkoumaného obrazu. Počet obrazů v okolí, podle kterých se predikce určují, je uživatelem

definovaná konstanta, případně se může měnit na základě místní hustoty bodů, což vede k analogii základního algoritmu. Nejčastější volbou pro měření vzdálenosti je použití Euklidovské vzdálenosti

$$||\mathbf{x}||_2 = \sqrt{\mathbf{x}_1^2 + \cdots + \mathbf{x}_n^2}, \tag{3.7}$$

kde n je počet obrazů, ale je možné použít obecně jakoukoli metrickou míru, která nejlépe odpovídá vstupnímu prostoru. Metody založené na sousedech jsou známé jako nezobecňující metody strojového učení, protože se nesnaží vytvořit obecný vnitřní model vstupního prostoru, ale jednoduše si „pamatují“ všechna svá trénovací data.

Klasifikátor k-nejbližších sousedů v `sklearn.neighbors.KNeighborsClassifier` je jednou z nejčastěji používaných metod klasifikace. Optimální volba hodnoty k je značně závislá na datech, a je to tedy parametr, který musí být vhodně zvolený. Obecně větší k potlačuje vliv šumu, ale způsobuje, že hranice klasifikace jsou méně zřetelné.

Základní klasifikace podle nejbližších sousedů používá jednotné váhy. To znamená, že hodnota přiřazená danému obrazu se vypočítá z prostého většinového hlasování nejbližších sousedů. Za určitých okolností je lepší vážit sousedy tak, aby bližší sousedé přispívali k přiřazení více. Tato váha lze zvolit jako stejnoměrná, podle vzdálenosti nebo je možné použít jakoukoliv jinou váhovou vzdálenostní funkci.

Navzdory své jednoduchosti se metody nejbližších sousedů osvědčily v mnoha klasifikačních a regresních problémech, včetně ručně psaných číslic a scén satelitních snímků. Jelikož se jedná o neparametrickou metodu, je často úspěšná v klasifikačních situacích, kdy je rozhodovací hranice velmi nepravidelná.

Rychlý výpočet nejbližších sousedů je aktivní oblastí výzkumu v oblasti strojového učení. Nejnaivnější implementace vyhledávání nejbližších sousedů zahrnuje hrubý výpočet vzdáleností mezi všemi dvojicemi bodů v souboru dat, tedy pro N vzorků v D dimenzích má tento přístup složitost $O[ND^2]$. Vyhledávání sousedů touto hrubou silou může být velmi konkurenceschopné pro malé vzorky dat. S rostoucím počtem vzorků N se však přístup hrubé síly rychle stává neproveditelným.

Výsledky predikce klasifikátor podporuje pravděpodobností, která je přímo poměrem počtu obrazů nejzastoupenější třídy vůči k nejbližším sousedům

$$P(y = j|X = x) = \frac{1}{k} \sum_{i \in \mathcal{C}} I(y^{(i)} = j), \tag{3.8}$$

kde $\mathcal{C}$ je množina značek tříd a funkce I vrací 1, když je její argument pravda, jinak je 0 [Pedregosa a kol., 2011].

3.3 Náhodný rozhodovací les

Náhodný les je kombinovaná učící metoda, která aplikuje několik klasifikátorů, rozhodovacích stromů, na různých částech datové sady a používá průměrování ke zlepšení přesnosti predikce a kontrole přetrénování.

Rozhodovací stromy

Rozhodovací stromy (angl. Decision Trees) jsou neparametrická metoda učení s učitelem, používaná pro klasifikaci a regresi. Cílem je vytvořit model, který predikuje hodnotu cílové proměnné tím, že z vlastností dat odvozuje jednoduchá rozhodovací pravidla. Rozhodovací strom obvykle začíná jedním uzlem, který se větví na možné výsledky. Každý z těchto výsledků vede k dalším uzlům, které se větví na další možnosti. Díky tomu má stromový tvar.
Výhodou rozhodovacích stromů je, že jsou:

- Jednoduché na pochopení a interpretaci. Stromy lze vizualizovat.
- Dosahují při predikci logaritmické složitosti vzhledem k počtu trénovacích obrazů.
- Dokáží zpracovávat problémy s více výstupy.
- Používají model „bílé skříňky“. Pokud je daná situace pozorovatelná v modelu, vysvětlení stavu je snadno vysvětlitelné pomocí booleovské logiky. Naproti tomu např. v umělé neuronové síti, jejíž model odpovídá „černé skříňce“, může být interpretace výsledků obtížnější.
- Umožňují ověřit spolehlivost modelu pomocí statistických testů.

Mezi nevýhody rozhodovacích stromů patří např:

- Mohou vytvářet při učení příliš složité stromy, které vedou k přetrénování. To je možné řešit prořezáváním, nastavením minimálního počtu požadovaných vzorků v listovém uzlu, nastavení maximální hloubky stromu.
- Mohou být nestabilní, protože malé změny v datech mohou vést k vytvoření zcela jiného stromu.
- Předpovědi rozhodovacích stromů nejsou ani hladké, ani spojité, ale po částech konstantní aproximace. Proto nemusí být dobré při extrapolaci.
- Vytvářejí při učení nevyvážených dat zkreslené stromy. Je pro ně tedy vhodné použít rovnoměrný počet obrazů ve třídách.

Pro danou množinu trénovacích obrazů $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, kde $\mathbf{x}_i \in \mathbb{R}^m$, $y_i \in \mathbb{N}^k$, m je velikost příznakového vektoru a k je počet tříd, rozdělí rozhodovací strom rekurzivně prostor příznaků tak, že jsou obrazy se stejnými značkami seskupeny dohromady.

Nechť jsou rozdělená data v uzlu u reprezentována jako Q_u. Ta jsou rozdělována rozhodovací funkcí $\theta = (j, t_u)$, která sestává z příznaku j a prahu t_u do podmnožin $Q_u^{leva}(\theta)$ a $Q_u^{prava}(\theta)$:

$$\begin{aligned} Q_u^{leva}(\theta) &= \{(\mathbf{x}, y) | \mathbf{x}_j \leq t_u\}, \\ Q_u^{prava}(\theta) &= Q_u \setminus Q_u^{leva}(\theta). \end{aligned} \tag{3.9}$$

Takto jsou podmnožiny $Q_u^{leva}(\theta)$ a $Q_u^{prava}(\theta)$ rekurzivně rozdělovány, dokud není dosaženo maximální povolené hloubky stromu nebo počtu listových uzlů.

Klasifikace pomocí náhodného rozhodovacího lesa

V klasifikátoru `sklearn.ensemble.RandomForestClassifier` je pro každý strom vymezena část trénovacích dat, která je sestavena pomocí náhodného výběru s vracením, tzv. *Bootstrapping* metodou. Při tomto sestavování jsou v posledním vybírání zvolené vzorky navráceny do původní množiny, takže mohou být náhodně vybrány znovu. Z těchto pod-částí trénovacích dat jsou sestaveny rozhodovací stromy, které jsou schopné dobře klasifikovat jen v pro ně známé části vstupního prostoru, ale na dobré klasifikování v celém prostoru jsou příliš „slabé“. Jejich spojením pomocí metody Gradient Boosting vznikne rozhodovací strom. Algoritmus se při dělení každého uzlu během konstrukce stromu snaží najít nejlepší rozdělovací funkce buď ze všech příznaků vstupního prostoru, nebo z náhodné podmnožiny o parametrem definované velikosti.

Tyto náhodnosti snižují rozptyl rozhodovacího lesu. Jednotlivé rozhodovací stromy totiž obvykle vykazují vysoký rozptyl, čímž přináší vlastní chyby predikcí a mají také tendenci k přetrénování. Jejich zprůměrováním v rámci rozhodovacího lesu lze některé z těchto chyb anulovat. Náhodné lesy dosahují snížení rozptylu kombinací různorodých stromů, někdy za cenu mírného zvýšení zkreslení. V praxi toho snížení rozptylu převyšuje míru zkreslení, což vede k celkově lepšímu modelu.

Metoda *Gradient Boosting* je technika strojového učení, která vytváří predikční model jako soubor slabých predikčních modelů, kterými jsou typicky rozhodovací stromy. Model vytváří postupně, podobně jako ostatní Boosting metody, a zobecňuje je tím, že provádí optimalizaci libovolné diferencovatelné ztrátové funkce. Rozhodovací les F_S je pomocí

metody Gradient Boosting sestrojen jako suma rozhodovacích stromů

$$F_S(\mathbf{x}_i) = \sum_{s=1}^{S} h_s(\mathbf{x}_i), \tag{3.10}$$

kde h_s je množina S slabých klasifikátorů. Výstup z funkce rozhodovacího stromu je ale spojitý, takže nepredikuje přímo třídu zařazení obrazu. Mapování z hodnoty $F_S(\mathbf{x}_i)$ na značku třídy y je závislé na použité ztrátové funkci. Pravděpodobnost, že obraz $\mathbf{x}_i$ patří do třídy y, je

$$P(y = 1|\mathbf{x}_i) = \sigma(F_S(\mathbf{x}_i)), \tag{3.11}$$

kde σ je sigmoidní funkce. Tím, že klasifikátor pracuje přímo s pravděpodobností, vrací tuto hodnotu spolu s predikcemi. Pro klasifikaci do více tříd se při každé iteraci S sestaví k stromů pro k tříd. Pravděpodobnost, že $\mathbf{x}_i$ patří do třídy k, se získá jako výsledek funkce softmax z hodnot $F_{S,k}(\mathbf{x}_i)$. Funkce softmax je definována jako

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}, \tag{3.12}$$

kde σ je označení softmax funkce, $\mathbf{z}$ je vstupní vektor, e^{z_i} je exponenciální funkce vstupního vektoru, e^{z_j} je exponenciální funkce výstupního vektoru a k je počet tříd pro klasifikaci [Pedregosa a kol., 2011].

3.4 Ridge klasifikátor

Ridge klasifikátor je postavený na *Ridge* regresi, což je model vycházející z metody nejmenších čtverců, který řeší některé její nedostatky penalizací koeficientů. Lineární funkce nejmenších čtverců je tedy pro regresní model ztrátovou funkcí a k penalizaci používá L2 normu:

$$\min_{\mathbf{w}} ||\mathbf{x}\mathbf{w} - y||_2^2 + \alpha||\mathbf{w}||_2^2, \tag{3.13}$$

kde parametr $\alpha \geq 0$ řídí míru smršťování (angl. shrinkage), což je způsob, jakým se extrémní hodnoty vzorku „smršťují“ směrem k centrální hodnotě, kterou může být jejich průměr. Čím větší je hodnota α, tím větší je míra smrštění.

Samotný Ridge klasifikátor nejprve převede výstupní třídy na binární hodnoty $\{-1, 1\}$ a poté s problémem zachází jako s regresní úlohou, přičemž optimalizuje stejný cíl jako je výše, tedy hledá optimální rozdělovací nadrovinu pro rozdělení tříd trénovacích dat:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} ||\mathbf{x}\mathbf{w} - y||_2^2 + \alpha||\mathbf{w}||_2^2. \tag{3.14}$$

Predikovaná třída odpovídá znaménku předpovědi regresoru. V případě klasifikace do více tříd se s problémem zachází jako s regresí s více výstupy a předpovídaná třída odpovídá výstupu s nejvyšší hodnotou.

Jako doplněk k predikcím poskytuje Ridge klasifikátor i hodnotu vzdálenosti obrazu $\mathbf{x}_i$ od rozdělovací nadroviny definované koeficienty $\mathbf{w}^*$. Někdy se také označuje jako *Least Squares Support Vector Machines s lineárním jádrem* [Pedregosa a kol., 2011].

3.5 Linear Support Vector klasifikátor

Klasifikátory s podpůrnými vektory (angl. Support Vector Machine, SVM) jsou sadou metod učení s učitelem, které se používají pro klasifikaci, regresi nebo detekci anomálií. Jejich výhodou je, že jsou:

- Efektivní ve velkorozměrových prostorech.
- Stále účinné v případech, kdy je počet dimenzí větší než počet vzorků.
- Využívají podmnožinu trénovacích bodů v rozhodovací funkci (tzv. podpůrné vektory), takže je také paměťově efektivní.
- Univerzální: pro rozhodovací funkci lze zadat různé kernelové funkce. K dispozici jsou běžné kernely, ale je možné použít i vlastní.

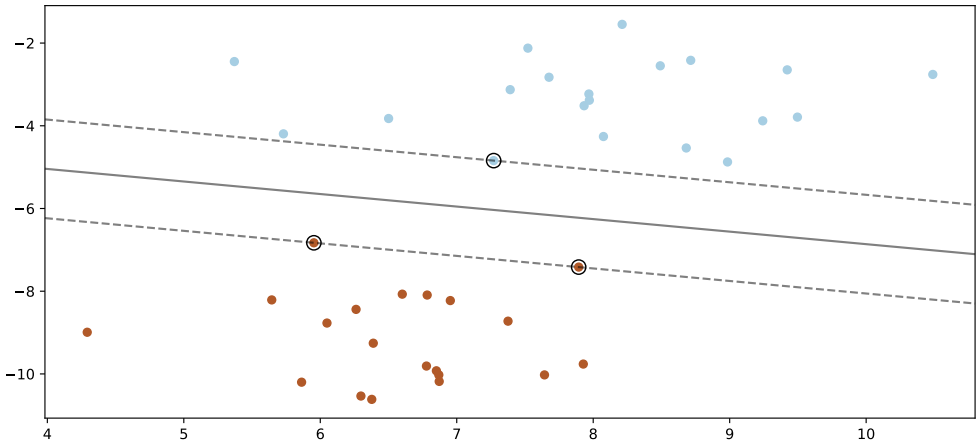
SVM konstruuje nadrovinu nebo sadu nadrovin ve vysokém nebo nekonečném rozměrovém prostoru, který lze použít pro klasifikaci, regresi nebo jiné úlohy. Intuitivně je dobré separace dosaženo nadrovinou, která má největší okraj, tedy vzdálenost k nejbližším tréninkovým obrazům libovolné třídy. Obecně platí, že čím větší je okraj, tím menší je celková chyba klasifikátoru. Na obrázku 3.1 je znázorněna rozhodovací funkce pro lineárně separovatelný problém se třemi vzorky na hranicích okrajů, které se nazývají *podpůrné vektory*.

Cílem při trénování algoritmu je tedy maximalizovat minimální vzdálenost obrazů tříd od rozdělovací nadroviny. Pro konkrétní obraz $\mathbf{x}_0$ je tato vzdálenost definována vztahem

$$d_H(\mathbf{x}_0) = \frac{|\mathbf{w}^T(\mathbf{x}_0) + b|}{||\mathbf{w}||_2}, \tag{3.15}$$

kde n-rozměrný vektor $\mathbf{w}$ a skalární hodnota b spolu definují rozdělovací nadrovinu, a $||\mathbf{w}||_2$ je euklidovská norma definovaná vztahem 3.7 a největší minimální vzdálenost je pak

$$\mathbf{w}^* = \arg\max_{\mathbf{w}}[\min_n d_H(\mathbf{x}_n)]. \tag{3.16}$$



Obr. 3.1: Ukázka rozdělení dvou separabilních shluků nadrovinou s podpůrnými vektory.

Ve druhé fázi je pak získaná rozdělovací nadrovina použita pro predikci tříd. Obraz $\mathbf{x}_n$ patří do třídy y_n, jestliže je jeho predikce $y_n[\mathbf{w}^T(\mathbf{x}_n) + b]$ větší než 0. Výraz

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \frac{\min_n y_n|\mathbf{w}^T(\mathbf{x}_0) + b|}{||\mathbf{w}||_2} \tag{3.17}$$

je pak základním vztahem pro klasifikaci obrazu do třídy pomocí SVM [Kunchhal, 2020].

Objekt `sklearn.svm.LinearSVC` je jednou z rychlejších implementací SVM klasifikace s lineárním kernelem. Vychází ze základního vztahu pro SVM klasifikaci, který je obohacený o regularizační parametr C, který reguluje vliv špatně zařazených obrazů z důvodu překryvu tříd a normalizovaný tak, aby velikost okrajů byla 1. Jeho primární problém může být formulován jako ekvivalent

$$\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||_2 + C\sum_{n=1} \max(0, y_n[\mathbf{w}^T(\mathbf{x}_n) + b]). \tag{3.18}$$

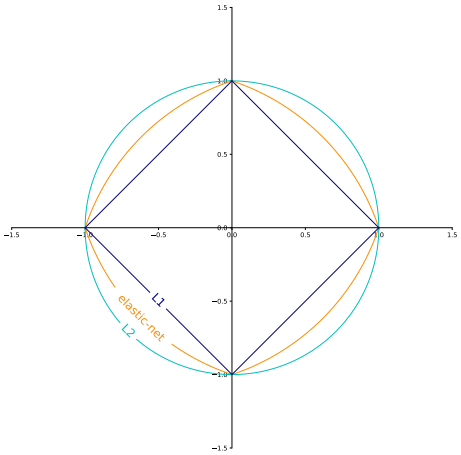
Výsledky predikce klasifikátor podporuje vzdáleností obrazu od rozdělovací nadroviny se znaménkem, které kladné znamená, že se obraz nachází v prostoru třídy, záporné že je obraz až za rozdělovací nadrovinou.

3.6 SVM s přístupem Stochastic Gradient Descent

Objekt `sklearn.linear_model.SGDClassifier` rozšiřuje výše popsaný SVM klasifikátor o přístup Stochastic Gradient Descent (SGD), což je jednoduchý, ale velmi účinný přístup k trénování lineárních klasifikátorů s konvexní ztrátovou funkcí, kterým např. SVM je. SGD byl úspěšně použit na rozsáhlé problémy s řídkými daty, které se často vyskytují při klasifikaci textu a zpracování přirozeného jazyka. Vzhledem k tomu, že data jsou řídká,

klasifikátory v tomto modulu snadno škálují na problémy s více než 10^5 trénovacími příklady a více než 10^5 příznaky. Přes to všechno je SGD pouze optimalizační technikou a nevytváří konkrétní rodinu algoritmů strojového učení. Je to pouze způsob, jak trénovat model.

Mezi výhody přístupu Stochastic Gradient Descent patří vysoká efektivita, snadná implementace a možnosti pro ladění. Nevýhodou je, že vyžaduje řadu hyperparametrů, jako je třeba parametr regularizace a počet iterací a je citlivý na škálování prvků.



Obr. 3.2: Zobrazení 3 variant penalizačních členů s penalizací = 1.

Matematicky je možné SGD popsat následujícím způsobem. Nechť $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ je množina trénovacích obrazů, kde $\mathbf{x}_i \in \mathbb{R}^m$ a $y_i \in -1, 1$ pro binární klasifikaci. Cílem je natrénovat parametry lineární skórovací funkce $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$. Předpovědí pro binární klasifikaci lze získat jednoduchým odečtením znaménko funkce $f(\mathbf{x})$. Pro nalezení parametry modelu $\mathbf{w} \in \mathbb{R}^m$ a $b \in \mathbb{R}$, je třeba minimalizovat regularizovanou trénovací chybu, která je dána vztahem

$$E(\mathbf{w}, b) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) + \alpha R(\mathbf{w}), \tag{3.19}$$

kde L je ztrátová funkce, která měří (ne)vhodné natrénování a R je penalizační člen, který penalizuje složitost modelu. α je pak nezáporný hyperparametr, který reguluje penalizaci. Všechny varianty ztrátové funkce ukazují horní hranici chyby chybné klasifikace a jako příklad ztrátové funkce je možné uvést *Hinge loss* s předpis $L(y_i, f(\mathbf{x}_i)) = \max(0, 1 - y_i, f(\mathbf{x}_i))$.

Parametr penalizace se často volí z následujících tří variant:

- L2 norma: $R(\mathbf{w}) := \frac{1}{2}\sum_{j=1}^{m} \mathbf{w}_j^2 = ||\mathbf{w}_j||_2^2$
- L1 norma: $R(\mathbf{w}) := \sum_{j=1}^{m} |\mathbf{w}_j|$
- Elastic Net: $R(\mathbf{w}) := \frac{\rho}{2}\sum_{j=1}^{n} \mathbf{w}_j^2 + (1+\rho)\sum_{j=1}^{m} |\mathbf{w}_j|$

Varianta penalizace Elastic Net je vlastně kombinací L2 a L1 normy, kde $\rho = 1 - l_1\_ratio$ a l_1_ratio určuje velikost jejich konvexní kombinace. Pro představu jsou všechny tři zobrazené na obrázku 3.2 [Pedregosa a kol., 2011].

4

Učení s částečnou informací od učitele

Učení s částečnou informací od učitele, angl. semi-supervised learning, je odnož strojového učení, která kombinuje učení s učitelem a učení bez učitele, popsané v předchozí kapitole. Většinou je použito v případech, kdy je cílem zvýšit úspěšnost jednoho z nich použitím informace, kterou poskytuje to druhé. Např. výsledek zařazení obrazu do třídy získané klasifikací neznámých dat může být použito pro následné vylepšení procesu klasifikace. Tedy použije se informace, která nebyla předem známa od učitele, jako taková, která se zařadí mezi informace, které od učitele byly na začátku procesu známy. Opačný přístup může být shlukovací algoritmus, založený na učení bez učitele, který vylepší proces shlukování pomocí informací o správném zařazení několika obrazů, které získá od učitele. Tak může provést sloučení nebo rozdělení některých shluků, které by bez takové informace provést nemohl.

Většina výzkumu v oblasti učení s částečnou informací od učitele se zaměřuje na klasifikaci. Není divu. Dnes je celkem levné sbírat data, ale drahé je nechat nějakého experta označkovat. Proces označkování dat správnou třídou zařazení je pro člověka náročným úkolem, a to zejména z časového hlediska, tedy trvá dlouhou dobu. Pro představu: Jak dlouho může trvat člověku označkovat tisíce nebo i řádově daleko více dat? V tomto právě můžou pomoct algoritmy, které zvýší úspěšnost klasifikátoru, na základě relativně málo dat označkovaných expertem, které použije k rozšíření trénovací množiny. Existují také algoritmy, které používají přístupy učení s částečnou informací od učitele, ale ne z důvodu málo označkovaných dat, ale pro samotné zlepšení klasifikátoru.

4.1 Předpoklady učení s částečnou informací od učitele

Pro metody, které mají být schopny se samy učit, resp. zlepšovat výsledky klasifikace, je nutné, aby data, se kterými pracuje, splňovala určité předpoklady. Nezbytnou podmínkou pro učení s částečnou informací od učitele je, že základní marginální rozdělení dat $P(\mathbf{x})$ ve vstupním prostoru obsahuje informace o aposteriorním rozdělení $P(y|\mathbf{x})$. Pokud tomu tak je, je možné použít neoznačená data k získání informací o $P(\mathbf{x})$, a tím i o $P(y|\mathbf{x})$. Pokud naopak tato podmínka splněna není a $P(\mathbf{x})$ neobsahuje žádnou informaci o $P(y|\mathbf{x})$, je ze své podstaty nemožné zlepšit přesnost předpovědí na základě dodatečných neoznačených dat [Zhu, 2008].

Naštěstí se zdá, že výše uvedená podmínka je ve většině problémech strojového učení, s nimiž se setkáváme v reálném světě, splněna. Ostatně, jinak by metody učení s částečnou informací od učitele nemohly být úspěšně nasazeny. Způsob interakce $P(\mathbf{x})$ a $P(y|\mathbf{x})$ však není vždy stejný. To dalo pro toto učení vzniknout předpokladům, které formalizují typy očekávaných interakcí [Chapelle a kol., 2006a]. Mezi nejrozšířenější předpoklady patří předpoklad hladkosti (angl. smoothness), nízké hustoty (angl. low-density) a varietní (angl. manifold) předpoklad. Tyto předpoklady jsou základem většiny, ne-li všech algoritmů učení s částečnou informací od učitele, které obecně závisí na splnění jednoho nebo více z nich, ať už explicitně, nebo implicitně.

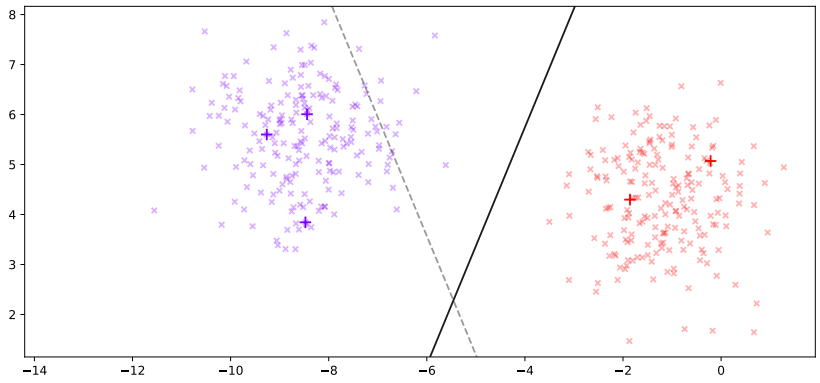
4.1.1 Předpoklad hladkosti

Předpoklad hladkosti říká, že pro dva vstupní body $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, které jsou si ve vstupním prostoru blízké platí, že jejich odpovídající označení y, y' by mělo být stejné. Tento předpoklad se běžně používá při učení s učitelem, ale velký přínos má jeho rozšíření na učení s částečnou informací od učitele. Např. je možné předpoklad o hladkosti aplikovat na neoznačená data. Nechť existuje označený datový bod $\mathbf{x}_1 \in \mathcal{X}_L$ a dva neoznačené datové body $\mathbf{x}_2, \mathbf{x}_3 \in \mathcal{X}_U$ tak, že $\mathbf{x}_1$ je blízko $\mathbf{x}_2$ a $\mathbf{x}_2$ je blízko $\mathbf{x}_3$, ale $\mathbf{x}_1$ není blízko $\mathbf{x}_3$. Pak vzhledem k předpokladu hladkosti můžeme stále očekávat, že $\mathbf{x}_3$ bude mít stejnou značku jako $\mathbf{x}_1$, protože blízkost - a tím i značka - se tranzitivně šíří přes $\mathbf{x}_2$.

4.1.2 Předpoklad nízké hustoty

Předpoklad nízké hustoty znamená, že rozdělovací nadrovina klasifikátoru by měla ve vstupním prostoru procházet primárně oblastmi, které mají nízkou hustotou. Jinými slovy, by neměla procházet oblastmi, kde je hustota vysoká. Předpoklad je definován

nad skutečným rozdělením vstupních dat $P(\mathbf{x})$. Při uvažování omezené množiny vzorků z tohoto rozdělení to v podstatě znamená, že rozhodovací hranice by měla ležet v oblasti, kde je pozorováno málo datových bodů. V tomto světle je předpoklad nízké hustoty úzce spjat s předpokladem hladkosti, a to tak, že jej lze v podstatě považovat za opak. Existuje-li oblast s nízkou hustotou, tj. oblast $R \subset \mathcal{X}$, kde je $P(\mathbf{x})$ nízké, pak se dá očekávat, že v oblasti R je obsaženo velmi málo obrazů, a je tedy nepravděpodobné, že by v oblasti R byla nalezena jakákoli dvojice podobných datových bodů. Pokud rozhodovací hranici umístíme do této oblasti s nízkou hustotou, předpoklad hladkosti není porušen, protože se týká pouze dvojic podobných datových bodů. U oblastí s vysokou hustotou lze naopak očekávat mnoho datových bodů. Umístění rozhodovací hranice do oblasti s vysokou hustotou tedy porušuje předpoklad hladkosti, protože predikované značky pro podobné datové body by pak nebyly podobné. Totéž platí i naopak: pokud platí předpoklad o hladkosti, pak jakékoli dva datové body, které leží blízko sebe, mají stejnou značku. Proto se dá očekávat,



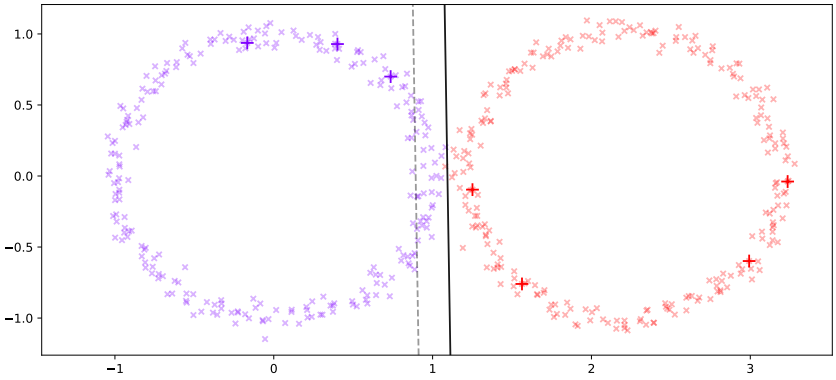
Obr. 4.1: Znázornění předpokladů hladkosti a nízké hustoty.

že v jakékoli hustě pokryté oblasti vstupního prostoru budou mít všechny datové body stejnou značku. V důsledku toho lze navrhnout rozdělovací rovinu tak, aby ve vstupním prostoru procházela pouze oblastmi s nízkou hustotou, čímž je splněn i předpoklad nízké hustoty. Světlé body na obr. 4.1 znázorňují neanotovaná a tmavé anotovaná data. Šedá čárkovaná čára pak rozdělovací nadrovinu natrénovanou pomocí klasifikátoru učení s učitele a černá rozdělovací nadrovina možné rozdělení klasifikátorem, který k učení využije i neanotovaná data. Hustota bodů ve shlucích také ukazuje oba výše uvedené předpoklady.

4.1.3 Varietní předpoklad

V úlohách strojového učení, kde lze data reprezentovat v euklidovském prostoru, se pozorované obrazy ve vysokodimenzionálním vstupním prostoru $\mathbb{R}^d$ jsou obvykle soustře-

děny podél nižších dílčích struktur. Tyto dílčí struktury se nazývají variety (angl. manifold). Jsou to topologické prostory, které jsou lokálně euklidovské. Například pro třírozměrný vstupní prostor, kde všechny body leží na povrchu koule, lze říci, že data leží na dvourozměrném mnohoúhelníku, viz znázornění varietního předpokladu na obr. 4.2, které je ve stejném duchu, jako předchozí obrázek 4.1. Varietní předpoklad v učení s částečnou informací od učitele říká, že a) vstupní prostor se skládá z více nižších rozměrů, na nichž leží všechny datové body a b) datové body ležící na stejné varietě patří do stejné třídy. Pokud jsme tedy schopni určit, které variety existují, a které obrazy leží na které varietě, lze přiřazení tříd neoznačených obrazů odvodit z označkovaných obrazů na stejné varietě [Engelen, Hoos, 2020].



Obr. 4.2: Znázornění varietního předpokladu.

4.2 Vhodnost využití

Výše popsané předpoklady jsou základem pro funkčnost učení s částečnou informací od učitele, jehož hlavním cílem je využít neoznačená data k vytvoření lepších postupů učení. Jak se ukazuje, ne vždy je to snadné nebo dokonce možné. Neoznačená data jsou užitečná pouze tehdy, pokud nesou informace užitečné pro predikci tříd, které nejsou obsaženy v samotných označených datech nebo je z nich nelze snadno získat. Aby bylo možné jakoukoli metodu učení s částečnou informací od učitele použít v praxi, musí být algoritmus schopen tyto informace extrahovat.

Bohužel se ukázalo, že najít praktickou odpověď na tuto otázku není jednoduché. Nejenže je obtížné přesně definovat podmínky, za kterých může jakýkoli konkrétní algoritmus učení s částečnou informací od učitele fungovat, ale také lze málokdy jednoduše vyhodnotit, do jaké míry jsou tyto podmínky splněny. Lze však uvažovat o použitel-

nosti různých metod učení na různé typy problémů. Například metody založené na grafech se obvykle spoléhají na lokální míru podobnosti při konstrukci grafu nad všemi datovými body. Pro úspěšné použití takových metod je důležité, aby bylo možné navrhnout smysluplnou lokální míru podobnosti. Ve vysokodimenzionálních datech, jako jsou obrázky, kde je euklidovská vzdálenost rysů zřídkakdy dobrým ukazatelem podobnosti mezi datovými body, je to často obtížné. Rozšíření algoritmů učení s učitelem na učení s částečnou informací od učitele naproti tomu obecně vychází ze stejného předpokladu jako jejich základní protějšky - algoritmy učení s učitelem. Například obě varianty algoritmu Support Vector Machines (SVM) se spoléhají na předpoklad nízké hustoty, který říká, že rozhodovací hranice by měla ležet v oblasti rozhodovacího prostoru s nízkou hustotou. Pokud v takových případech dobře funguje klasifikátor, který má všechna data označkovaná učitelem, je přirozené použít jeho rozšíření na částečnou informaci od učitele. Stejně jako v případě algoritmů učení s učitelem nebyla dosud objevena žádná metoda, která by apriori určila, jaká metoda učení je pro konkrétní problém nejvhodnější. Navíc nelze zaručit, že zavedení neoznačených dat nezhorší výkonnost. Takové zhoršení výkonnosti bylo v praxi pozorováno v několika studiích [Zhu, 2008]; [Chapelle a kol., 2009]. To je zvláště důležité ve scénářích, kde lze dosáhnout velmi dobrého výkonu již s klasifikátory trénovanými s učitelem. V těchto případech je potenciální zhoršení výkonu mnohem větší než jeho potenciální nárůst.

Hlavním závěrem z těchto pozorování je, že učení s částečnou informací od učitele by nemělo být považováno za zaručený způsob, jak dosáhnout lepšího výkonu predikce pouhým zavedením neoznačených dat. Spíše by se k němu mělo přistupovat jako k dalšímu směru v procesu hledání a konfigurace učícího algoritmu pro danou úlohu. Postupy učení s částečnou informací od učitele by měly být součástí souboru algoritmů zvažovaných pro použití v konkrétním aplikačním scénáři a k výběru přístupu, který je pro danou situaci vhodný, by měla být použita kombinace teoretické analýzy (pokud je to možné) a empirického hodnocení.

4.2.1 Empirické hodnocení metod

Ačkoliv je mnohdy náročné hodnotit metody učení s částečnou informací od učitele, případně míru splnění předpokladů, je možné metody hodnotit empiricky, tedy na základě konkrétních dat. Relativní úspěšnost jednotlivých algoritmů strojového učení je ovlivněna řadou rozhodnutí. V případě učení s učitelem k nim patří výběr datových sad, jejich rozdělení na trénovací, testovací a validační sady a míra vyladění hyperparametrů.

V případě učení s částečnou informací od učitele vstupují do hry další faktory. Nejprve je třeba rozhodnout, které obrazy mají být označkovány, a které mají zůstat bez označení. Dále, zda se bude úspěšnost učícího se algoritmu hodnotit na neanotovaných datech použitých pro trénování (což je z definice případ transduktivního učení), nebo na zcela nesouvislém souboru testů. Kromě toho je důležité stanovit kvalitní pravidla pro správné použití neanotovaných dat.

Jak bylo zjištěno v praxi, výběr datových sad a jejich rozdělení může mít významný vliv na relativní výkonnost různých učebních algoritmů, viz např. [Chapelle a kol., 2006b]; [Triguero a kol., 2015]. Některé algoritmy mohou fungovat dobře, když je množství anotovaných dat omezené, a špatně, když je k dispozici více anotovaných dat. Jiné mohou vynikat na určitých typech datových sad, ale ne na jiných. Pro realistické hodnocení algoritmů učení s částečnou informací od učitele je potřeba, aby vědci hodnotili své algoritmy na rozmanitých sadách dat s různým množstvím anotovaných a neanotovaných dat. Na závislost úspěšnosti konkrétních algoritmů na použitých datech ukazují studie, které hodnotily výkonnost různých metod na různých souborech dat. V nich bylo empiricky porovnáno jedenáct různých algoritmů, přičemž jako základní byly použity SVM s učitelem a kNN, za použití Semi-Supervised Support Vector Machines (3SVM), label propagation a techniky manifold regularization. Pro každý algoritmus byla také použita optimalizace hyperparametrů. Toto empirické porovnání vedlo ke zjištění, že na osmi různých souborech dat žádný algoritmus výrazně nepředčil ostatní. Na některých datových sadách bylo pozorováno výrazné zlepšení výkonu oproti základním algoritmům, zatímco na jiných byl zjištěn zhoršený výkon. Relativní výkon se také lišil v závislosti na množství neoznačených dat [Chapelle a kol., 2006b].

Kromě výběru datových sad a jejich rozdělení je při hodnocení výkonnosti metody učení s částečnou informací od učitele důležité zvolit pevnou výchozí úroveň. Koneckonců, v praxi není nijak zvlášť důležité, zda zavedení neanotovaných dat zlepší výkonnost nějakého jednoho konkrétního učícího se algoritmu. Hlavní otázka spíše zní, zda zavedení neanotovaných dat přináší celému učícímu se subjektu takové zlepšení, které mu zajistí lepší výkon, než jakémukoliv jinému učícímu se subjekt - ať už s úplnou nebo s částečnou informací od učitele. Při hodnocení výkonnosti algoritmů učících se s částečnou informací od učitele to vyžaduje zahrnutí nejmodernějších, řádně vyladěných kontrolovaných výchozích dat [Oliver a kol., 2018].

5

Metody učení s částečnou informací od učitele

Rozmach metod učení s částečnou informací od učitele je otázkou zejména posledních dvou desetiletí. Vědci přichází s různými přístupy práce s neanotovanými daty, které se liší v použití předpokladů, konkrétním využití neanotovaných dat a v jejich vztahu k učení s učitelem. Metody je možné rozdělit do dvou hlavních kategorií na induktivní a transduktivní, které se liší optimalizačním přístupem [Engelen, Hoos, 2020]. Induktivní metody se snaží najít klasifikační model, zatímco transduktivní se zabývají zejména získáním predikcí tříd pro neanotovaná data a žádný klasifikační model neposkytují. Jinými slovy, při zadání datové sady sestávající z anotovaných a neanotovaných dat $\mathbf{x}_L, \mathbf{x}_U \subseteq \mathcal{X}$, se štítky $\mathbf{y}_L \in \mathcal{Y}^l$ pro l označkovaných obrazů, induktivní metody poskytují model $f : \mathcal{X} \mapsto \mathcal{Y}$, zatímco transduktivní metody vytvářejí předpovědi značek $\hat{y}_U$ pro neoznačené obrazy v $\mathbf{x}_U$. Induktivní metody tedy zahrnují optimalizaci nad sestavenými predikčními modely, zatímco transduktivní metody optimalizují přímo nad predikcemi $\hat{y}_U$.

5.1 Induktivní metody učení

Induktivní metody obecně rozšiřují algoritmy učení s učitelem o neanotovaná data. Mohou tato neanotovaná data zahrnout buď ve fázi předzpracování, v cílové funkci nebo prostřednictvím pseudoanotování. Cílem induktivních metod je vytvořit klasifikátor, který dokáže generovat předpovědi pro libovolný objekt v oblasti vstupního prostoru. Při trénování tohoto klasifikátoru mohou být neanotovaná data anotována na základě informace z anotovaných dat. Pro další použití získaných pseudoanotovaných dat jsou

jejich označení považována za nezávislá vzhledem k původním anotovaným datům. Je zde velká podobnost v přístupu s metodami učení s učitelem. Model je vytvořen ve fázi trénování a poté může být použit pro predikci tříd nových obrazů.

5.1.1 Obalovací (Wrapper) metody

Obalovací metody patří mezi nejstarší a nejznámější algoritmy učení s částečnou informací od učitele [Zhu, 2008]. Jednoduchý přístup rozšíření stávajících algoritmů s učitelem na učení s částečnou informací od učitele spočívá v tom, že jeden nebo více klasifikátorů se natrénuje nejprve na anotovaných datech, a poté se výsledné predikce neanotovaných dat použijí k vytvoření dalších anotovaných dat. Tato data se běžně označují jako automaticky anotovaná nebo *pseudoanotovaná data*. Postup se obvykle skládá ze dvou střídajících se fází trénování a pseudoanotování. V trénovací fázi se na anotovaných a případně na pseudoanotovaných datech z předchozích iterací trénuje jeden nebo více klasifikátorů učících se s učitelem. Ve fázi pseudoanotace jsou výsledky klasifikací použity k odvození tříd pro dříve neoznačkované obrazy. Z obrazů, u kterých si byly klasifikátory „nejvíce jisté“ svými predikcemi, je jejich pseudooznačení použito v další iteraci.

Významnou výhodou obalovacích metod je, že je lze použít prakticky s jakoukoli základní metodou učení s učitelem. Taková metoda se chová standardně, jako kdyby šlo o běžné učení s učitelem a kolem ní je postavený algoritmus tzv. obálky, který základní metodě předává anotované a pseudoanotované obrazy, jako by se v obou případech jednalo o standardní anotovaná data. Některé obalovací metody vyžadují, aby základní učící se algoritmus poskytoval pravděpodobnostní předpovědi, nicméně mnoho obalovacích metod, které se spoléhají na více základních učících se algoritmů, tak nečiní. Každá konkrétní obalovací metoda závisí na předpokladech učení s částečnou informací od učitele.

Obalovací metody se dále rozdělují podle počtu použitých klasifikátorů, zda-li jsou použité různé nebo stejné typy klasifikátorů a jestli používají data s jedním nebo více pohledy, tj. zda jsou data rozdělena na více podmnožin příznaků.

5.1.1.1 Metoda samoučení

Metody samoučení, nebo-li učení sebe sama, patří mezi nejzákladnější přístupy k pseudoanotaci [Triguero a kol., 2015]. Skládají se z jediného klasifikátoru, který je iterativně trénován jak na anotovaných, tak na pseudoanotovaných datech z předchozích iterací. Samoučení poprvé navrhl David Yarowsky jako přístup k rozklíčování významu slov v textových dokumentech, kdy předpovídá význam slov na základě jejich kon-

textu [Yarowsky, 1995]. Od té doby bylo navrženo několik aplikací a variant samoučení, například aplikace na problémy detekce objektů, kde byla prokázána lepší výkonnost oproti tehdy nejmodernějšímu přímému modelu detekce objektů [Rosenberg a kol., 2005].

Na začátku procesu trénování se klasifikátor trénuje pouze na datech anotovaných od učitele. Jeho výsledky predikcí se použijí k získání věrohodných pseudoanotovaných dat, která jsou pro příští iteraci přidána do sady anotovaných dat, na kterých je klasifikátor znovu trénován. Pro další iterace se tedy k původním anotovaným datům od učitele přidají nově získaná pseudoanotovaná data. Tento postup se obvykle opakuje, dokud nezůstanou žádná neoznačená data.

Při návrhu samoučící se metody je nutné počítat se spoustou parametrů, které je potřeba nastavit. Zejména se jedná o strategii výběru pseudoanotovaných dat, jejich opakovaného použití v pozdějších iteracích algoritmu a zastavovací kritérium. Strategie výběru pseudoanotovaných dat, která mají být použita v příští iteraci, je obzvláště důležitý, protože na ní závisí učení základního klasifikátoru. Typicky se tento výběr provádí na základě důvěryhodnosti predikce. Tato důvěryhodnost kvality odhadů pak významně ovlivňuje výkonnost algoritmu. Algoritmy, které nativně nepodporují robustní pravděpodobnostní předpovědi, mohou vyžadovat úpravy, aby mohly být k samoučení využity. Příkladem může být použití míry založené na lokální vzdálenosti obrazů [Tanha a kol., 2015].

5.1.1.2 Metoda kooperativního učení

Oproti metodě samoučení je kooperativní učení rozšířeno o jeden nebo více klasifikátorů, které se učí společně nebo jinak využívají vzájemnou informaci o klasifikování neanotovaných dat ke zvýšení úspěšnosti celého algoritmu.

. Všechny klasifikátory využívají k trénování stejnou množinu dat a jejich společné výsledky určují, která pseudoanotovaná data budou použita v další iteraci. Pro úspěšné kooperativní trénování je vhodné, aby základní učící se algoritmy nebyly ve svých předpovědích příliš silně korelovány. Pokud tomu tak je, jejich potenciál poskytovat si navzájem užitečné informace je omezený. V literatuře se tato podmínka obvykle označuje jako kritérium diverzity [Wang, Zhou, 2010]. Metoda kooperativního učení se dále rozděluje na použití stejného nebo různého pohledu na data. Existuje-li nebo je-li možné data rozdělit tak, aby na ně existovalo více pohledů, je vhodné jich pro zajištění kriteria diverzity využít. Například při práci s audiovizuálními daty lze data přirozeně rozložit na obrazovou a zvukovou složku. V mnoha reálných úlohách ale nejsou apriori odlišné pohledy na data možné.

Kooperativní učení s více pohledy

Algoritmy kooperativního učení s více pohledy se opírají o dva hlavní předpoklady: 1. každá jednotlivá podmnožina příznaků by měla být dostatečná k získání dobrých předpovědí na daném souboru dat a 2. podmnožiny příznaků by měly být podmíněně nezávislé vzhledem k označení třídy. První předpoklad lze chápat triviálně. Pokud jedna ze dvou podmnožin příznaků nestačí k vytvoření dobrých předpovědí, klasifikátor využívající tuto množinu nemůže nikdy pozitivně přispět k celkovému výkonu kombinovaného přístupu. Druhý předpoklad souvisí s kritériem diverzity. Pokud jsou podmnožiny příznaků vzhledem k označení třídy podmíněně nezávislé, je nepravděpodobné, že by byly predikce jednotlivých klasifikátorů silně korelované. Formálně, pro každý obraz $\mathbf{x}_i = \mathbf{x}_i^{(1)} \times \mathbf{x}_i^{(2)}$, rozložený na podmnožiny příznaků $\mathbf{x}_i^{(1)}$ a $\mathbf{x}_i^{(2)}$, platí předpoklad podmíněné nezávislosti $p(\mathbf{x}_i^{(1)}|\mathbf{x}_i^{(2)}, y_i) = p(\mathbf{x}_i^{(1)}|y_i)$.

V praxi není druhý předpoklad obecně splněn. I v případech, kdy přirozené rozdělení příznaků existuje, je nepravděpodobné, že by informace obsažené v jednom pohledu neposkytovaly žádné informace o pohledu druhém, pokud jsou podmíněny označením třídy. Pro představu, při klasifikaci novinových článků lze očekávat, že titulek, jakožto jeden pohled na data, bude obsahovat vodítka k obsahu samotného článku, jakožto druhý pohled. Pokud má článek titulek např. „SVM - mocný nástroj pro klasifikaci vašich dat“, je pravděpodobnější, že v textu článku budou informace o klasifikátoru SVM, než o jiném algoritmu strojového učení. Pomocí empirických metod bylo ukázáno, že pro úspěšné společné vzdělávání stačí slabý předpoklad nezávislosti, byť stále platí, že čím větší je nezávislost datových pohledů, tím více informace z nich můžou klasifikátory získat [Abney, 2002].

Kooperativní učení s jedním pohledem

Kooperativní trénování může být úspěšné i v případě, kde neexistuje použitelné rozdělení dat na více pohledů. Jeden pohled na data tak může být distribuován do více různých klasifikátorů nebo můžou být klasifikátory zařazeny za sebe, kdy se v každé iteraci natrénuje právě jeden klasifikátor. Díky tomu dojde k obohacení trénovací množiny, kterou v následující iteraci použije pro učení jiný klasifikátor. Takto se dokola střídají použité klasifikátory, až do ukončení běhu algoritmu.

Při použití více klasifikátorů, které se učí souběžně je možné využít i paralelního trénování, čímž se snížení časová náročnost algoritmu. Ačkoliv všechny klasifikátory využívají stejná data se stejným pohledem, můžou se lišit svojí úspěšností klasifikace.

V okolí obrazů, u kterých kde se výsledky jejich zařazení neshodují, je možné očekávat oblast, kde se data vstupního prostoru rozdělují mezi třídami. Pokud platí předpoklad nízké hustoty (4.1.2), měla by takových obrazů být ve zmíněné oblasti menšina.

V případech, kdy je to možné, se vědci snaží jedno-pohledová data uměle rozdělit na data s více pohledy, jelikož se ukazuje, že více pohledů na data mívá kladný vliv na kooperativní učení. V takovém případě je snahou vytvořit tolik pohledů, kolik je použitých klasifikátorů [Wang, Zhang, 2008].

5.1.1.3 Metoda Boosting

Termín boosting označuje rodinu algoritmů, které jsou schopny převést slabé klasifikátory na silné. Intuitivně je slabý klasifikátor jen o něco lepší než náhodný odhad, zatímco silný klasifikátor je velmi blízko perfektní bezchybné klasifikaci. Myšlenka boosting algoritmů je odpovědí na otázku, jestli je potenciálně možné jakýkoli slabě se učící systém posílit na silně se učící. Zejména, jestliže je v reálné praxi obecně velmi snadné získat slabý klasifikátor, ale obtížné získat klasifikátor silný [Zhou, 2012].

Obecný boosting postup je poměrně jednoduchý. Nechť existuje slabý klasifikátor, který pracuje na libovolném zadaném binárním rozdělení dat, tedy klasifikace případů jako pozitivní a negativní. Tréninkové instance v prostoru $\mathcal{X}$ jsou $i.i.d.$ vybírány z rozdělení $\mathcal{D}$ a základní pravdivostní funkce je f. Nechť prostor $\mathcal{X}$ se skládá ze tří částí $\mathcal{X}_1$, $\mathcal{X}_2$ a $\mathcal{X}_3$, kde každá je $1/3$ z celkového množství dat a klasifikátor pracující pouze s náhodným odhadem má v této úloze 50% chybu klasifikace. Cílem je získat přesný klasifikátor (např. s nulovou chybou), ale je k dispozici pouze slabý klasifikátor, který správně klasifikuje pouze v prostorech $\mathcal{X}_1$ a $\mathcal{X}_2$ a špatně v $\mathcal{X}_3$, má tedy $1/3$ klasifikační chybu. Takovýto slabý klasifikátor s označením h_1 je nedostatečný, a tedy h_1 není žádoucí.

Myšlenkou boosting algoritmu je „posílit“ původní klasifikátor h_1 a opravit jeho chyby. Je možné se pokusit z $\mathcal{D}$ odvodit nové rozdělení $\mathcal{D}'$, které chyby h_1 zviditelní, např. se více zaměří na případy v $\mathcal{X}_3$. Pak je možné z $\mathcal{D}'$ natrénovat klasifikátor h_2. Může se stát, že nový h_2 bude také slabý klasifikátor, který má správné klasifikace v $\mathcal{X}_1$ a $\mathcal{X}_3$ a má chybné klasifikace v $\mathcal{X}_2$. Vhodnou kombinací h_1 a h_2 bude mít kombinovaný klasifikátor správné klasifikace v $\mathcal{X}_1$ a možná nějaké chyby v $\mathcal{X}_2$ a $\mathcal{X}_3$.

Posílení prvního klasifikátoru druhým vylepšilo celkovou úspěšnost, ale ta je stále vzdálena požadovanému cíli. Nechť je tedy odvozeno další nové rozdělení $\mathcal{D}''$, kde jsou zřetelné chyby kombinovaného klasifikátoru, a z tohoto rozdělení je natrénován klasifikátor h_3 tak, aby h_3 správně klasifikoval v $\mathcal{X}_2$ a $\mathcal{X}_3$. Kombinací h_1, h_2 a h_3 pak získáme nejlepší klasifikátor, protože v každém prostoru $\mathcal{X}_1$, $\mathcal{X}_2$ a $\mathcal{X}_3$ provedou

správnou klasifikaci alespoň dva klasifikátory. Stručně řečeno, boosting funguje tak, že se postupně trénují sady klasifikátorů, které se kombinují pro predikci, přičemž pozdější klasifikátory se více zaměřují na chyby dřívějších klasifikátorů.

5.1.2 Předzpracování dat bez informace od učitele

Druhou kategorií induktivních metod je předzpracování dat bez informace od učitele. Ta většinou používá anotovaná a neanotovaná data odděleně. Zpravidla nejprve použije anotovanou část, ke které následně přidá neanotovaná data.

Extrakce informativních příznaků

Extrakce příznaků (angl. feature extraction) hraje důležitou roli při konstrukci klasifikátorů již od počátků strojového učení. Metody extrakce příznaků se snaží najít takovou transformaci vstupních dat, která zlepší výkon klasifikátoru nebo jeho konstrukci učiní výpočetně efektivnější.

Mnoho metod extrakce příznaků pracuje bez informace od učitele. Například *analýza hlavních komponent* (PCA) transformuje vstupní data na jinou bázi tak, aby byla lineárně nekorelovaná, a seřadí hlavní komponenty na základě jejich variance. Jiné tradiční algoritmy extrakce příznaků pracují s označenými daty a snaží se extrahovat příznaky s vysokou prediktivní silou.

Jsou případy, kdy data nejsou ze své podstaty reprezentována jako smysluplný vektor příznaků. Protože mnoho běžných klasifikačních metod takovou reprezentaci vyžaduje, je v těchto případech extrakce příznaků nutností. Krok extrakce příznaků tedy spočívá v nalezení vektorového prostoru, které odpovídá vztahům mezi jednotlivými vstupními objekty. Příkladem takové transformace pro zpracování přirozeného jazyka může být reprezentace dokumentů Bag-of-words, viz sekce 6.3.1.

Metoda cluster-then-label

Shlukování a klasifikace jsou tradičně považovány za relativně oddělené oblasti výzkumu. Mnoho algoritmů učení s částečnou informací od učitele však využívá principy shlukování k řízení procesu klasifikace. Přístupy založené na shlukování a následném označování tvoří skupinu metod, které explicitně spojují procesy shlukování a klasifikace. Nejprve aplikují algoritmus shlukování bez nebo s částečnou informací od učitele na všechna dostupná data a výsledné shluky použijí ke klasifikaci.

Jedním z přístupů k metodě cluster-then-label je, že algoritmus vytvoří nejprve shluky z anotovaných dat a z podmnožinu neanotovaných dat. Poté jsou na anotovaných datech natrénovány klasifikátory nezávisle pro každý shluk, pomocí nichž jsou neanotované obrazy klasifikovány. Tento přístup podporuje libovolný základní metodu učení s učitelem.

Dalším přístupem je aplikace samoorganizující se mapy na anotovaná data, která vytvoří anotované oblasti, kam jsou neanotovaná data přiřazována. Pokud shluk, do kterého je neoznačený datový bod $\mathbf{x}_i$ mapován, obsahuje pouze datové body se stejnou značkou, je tato značka přiřazena i bodu $\mathbf{x}_i$. Při mapování jsou upřednostňovány shluky s nízkou nečistotou značek, tedy vysokou mírou konzistence označkovaných obrazů v rámci daného shluku. Tento proces lze opakovat, dokud nejsou označkována všechna neanotovaná data. Tento přístup lze považovat za obalovací metodu (viz sekce 5.1.1).

5.2 Transduktivní metody

Na rozdíl od induktivních metod nekonstruují transduktivní metody klasifikátor pro celý vstupní prostor. Z toho vyplývá, že informace se musí šířit přímými spojeními mezi obrazy. Toto pozorování přirozeně dává podnět ke grafovému přístupu. Lze-li definovat graf, v němž jsou podobné datové body propojeny, lze pak informace šířit podél hran tohoto grafu. Existují i induktivní metody založené na grafech a v obou případech jsou obvykle založeny na varietním předpokladu, tedy konstruované na základě lokální podobnosti mezi obrazy. Transduktivní metody založené na grafech se obecně skládají ze tří kroků: konstrukce grafu, vážení grafu a odvozování. Protože metody učení s učitelem mají z definice přístup k neoznačkovaným datům až ve fázi testování, neexistují žádné jasné analogie transduktivních algoritmů v algoritmech učení s učitelem.

Transduktivní metody nerozlišují mezi fází učení a fází testování. Jsou jim poskytnuta anotovaná data $(\mathbf{x}_L, y_L)$ a neanotovaná data $\mathbf{x}_U$ a výstupem jsou výhradně predikce $\hat{y}_U$ neanotovaných dat. Transduktivní metody obvykle definují graf nad všemi obrazy, označenými i neoznačenými, který zaznamenává jejich párovou podobnost s případně váženými hranami. Poté se definuje a optimalizuje objektivní funkce tak, aby se dosáhlo dvou cílů:

1. Pro označkované obrazy by se predikované značky měly shodovat se skutečnými značkami.
2. Podobné obrazy, definované prostřednictvím grafu podobnosti, by měly mít stejné predikce značek.

Jinými slovy, tyto metody podporují konzistentní predikce pro podobné obrazy, přičemž berou v úvahu známé značky.

V prvním kroku jsou uzly, které představují jednotlivé obrazy, v grafu vzájemně propojeny na základě zvolené míry podobnosti. Ve druhém kroku jsou výsledné hrany váženy, čímž vznikne váhová matice. První dva kroky se společně označují jako fáze konstrukce grafu. Po konstrukci grafu máme graf sestávající z množiny uzlů $V = v_1, \ldots, v_n$, a váhové matice $W \in \mathbb{R}^{n \times n}$ obsahující váhy hran pro všechny dvojice uzlů. Nulová váha hrany znamená, že mezi uzly hrana neexistuje.

Jakmile je graf sestrojen, použije se k získání predikcí $\hat{y}_U$ pro neoznačkované uzly. Obecný tvar funkce pro transduktivní metody založené na grafech obsahuje jednu složku pro penalizaci predikovaných značek, které neodpovídají skutečnému označení, a další složku pro penalizaci rozdílů v predikcích značek pro spojené uzly. Formálně vzato, při zadání řízené ztrátové funkce ℓ pro anotovaná data a neřízené ztrátové funkce ℓ_U pro dvojice označkovaných nebo neoznačkovaných uzlů se transduktivní metody založené na grafech snaží najít takové označení $\hat{y}$, které minimalizuje vztah:

$$\lambda \cdot \sum_{i=1}^{l} \ell(\hat{y}_i, y_i) + \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} \cdot \ell_U(\hat{y}_i, \hat{y}_j), \tag{5.1}$$

kde λ řídí relativní důležitost členu, který disponuje informací od učitele. Některé metody založené na grafech navíc ukládají další unární regulární člen na neoznačené předpovědi. Tento obecný rámec pro metody založené na grafech umožňuje mnoho variant v každém ze svých kroků, nicméně uvedená formulace je u metod založených na grafech běžná.

Při konstrukci grafu se většina metod založených na grafech spoléhá na lokální podobnost mezi jednotlivými obrazy vstupního prostoru a spojuje takové, které mají podobné vlastnosti. V takovém případě se kromě varietního předpokladu implicitně spoléhají na předpoklad hladkosti. Metody založené na grafech berou v úvahu také podobnost mezi dvojicemi neoznačených uzlů. Na základě těchto informací lze značky šířit z označeného uzlu na neoznačený uzel tranzitivně přes další uzly, a to jak označené, tak neoznačené a získat tak predikce pro původně neanotovaná data. V tomto duchu lze některé metody založené na grafech považovat za rozšíření metod klasifikace pomocí k-nejbližších sousedů na učení s částečnou informací od učitele.

6

Klasifikace dokumentů

Klasifikace dokumentů, jinak známá také jako kategorizace textu nebo vyhledávání témat, je stále více využívanou úlohou strojového učení. Jde o označování textů v přirozeném jazyce tematickými kategoriemi z předem definované sady. První snahy o klasifikaci dokumentů se datují na počátek 60. let, ale teprve počátkem 90. let se díky zvýšenému aplikačnímu zájmu a dostupnosti výkonnějšího hardwaru stala významnou podoblastí oboru informačních systémů. Nyní se řešení klasifikace textu používá v mnoha kontextech, od indexování dokumentů na základě řízeného slovníku, přes filtrování dokumentů, automatizované generování metadat, tvorbu hierarchických slovníků webů, až po automatizované generování metadat a obecně v jakýchkoli aplikacích vyžadujících organizování dokumentů nebo jejich selektivní a adaptivní správu. Až do konce 80. let byl nejpopulárnějším přístupem ke klasifikaci dokumentů, alespoň v reálných aplikacích, přístup znalostního inženýrství, spočívající v ručním definování souboru pravidel, v nichž jsou zakódovány expertní znalosti o tom, jak klasifikovat dokumenty v rámci daných kategorií. V 90. letech tento přístup stále více ztrácel na popularitě ve prospěch paradigmatu strojového učení, podle něhož je možné obecným induktivním postupem automaticky vytvořit klasifikátor textu tím, že se ze souboru předem klasifikovaných dokumentů učí charakteristiky jednotlivých kategorií. Výhodou tohoto přístupu je přesnost srovnatelná s přesností dosahovanou lidskými experty a značná úspora expertní pracovní síly, neboť pro konstrukci klasifikátoru nebo pro jeho přenesení na jinou sadu není třeba zásahu znalostních inženýrů ani doménových expertů [Sebastiani, 2001].

Technikami klasifikace dokumentů lze obecně řešit mnoho problémů týkajících se organizace a archivace dokumentů, ať už pro účely osobní organizace nebo strukturování firemní dokumentové nebo korespondenční základny. Například v redakci novin musí být příchozí inzeráty před zveřejněním roztříděny do kategorií, jako jsou Osobní inzeráty, Auta

na prodej, Nemovitosti atd. Noviny, které se zabývají velkým objemem inzerátů, by automatický systém, který by pro daný inzerát vybral nejvhodnější kategorii, mohl usnadnit značné množství ruční, a hlavně monotónní práce. Dalšími možnými aplikacemi jsou uspořádání patentů do kategorií pro usnadnění jejich vyhledávání [Larkey, 1999], automatické zařazování novinových článků do příslušných rubrik (např. Politika, Domácí zpravodajství, Životní styl atd.), automatické seskupování konferenčních příspěvků do sekcí nebo automatické třídění příchozích e-mailů firmy správným lidem k jejich vyřízení, včetně filtrace spamu, či pošty, která nepotřebuje reakci (např. automatické odpovědi v nepřítomnosti).

6.1 Klasifikace textu

Obecně je klasifikace textu úlohou přiřazující logické hodnoty (T / F) každé dvojici $(d_i, c_i) \in \mathcal{D} \times \mathcal{C}$, kde $\mathcal{D}$ je množinou dokumentů a $\mathcal{C} = \{c_1, \ldots, c_k\}$, kde k je počet kategorií, je množina předem definovaných kategorií, nebo-li tříd. Hodnota T přiřazená (d_j, c_i) znamená rozhodnutí zařadit d_j pod c_i, zatímco hodnota F znamená rozhodnutí d_j pod c_i nezařadit. Formálněji řečeno, úkolem je aproximovat neznámou cílovou funkci $f : \mathcal{D} \times \mathcal{C}\{T, F\}$, nebo-li klasifikátor, která popisuje, jak by měly být dokumenty klasifikovány. Pro úlohu klasifikace jsou třídy pouze symbolické značky a není pro ně k dispozici žádná další znalost jejich významu. Pro základní klasifikaci dokumentů dále není k dispozici žádná exogenní znalost, tedy údaje poskytnuté pro účely klasifikace z externího zdroje. Klasifikace proto musí být provedena pouze na základě endogenní znalosti, tedy získané z dokumentů. To zejména znamená, že se nepředpokládá dostupnost metadat, jako je například datum nebo zdroj zveřejnění, typ dokumentu atd.

Spoléhat se pouze na endogenní znalosti znamená klasifikovat dokument pouze na základě jeho sémantiky a vzhledem k tomu, že sémantika dokumentu je subjektivní pojem, vyplývá z toho, že o příslušnosti dokumentu ke kategorii nelze rozhodnout deterministicky. Příkladem toho je fenomén nekonzistence mezi anotátory [Cleverdon, 1988]: když dva lidští experti rozhodují, zda zařadit dokument d_j do kategorie c_i, mohou se neshodnout, a to se ve skutečnosti děje s poměrně vysokou frekvencí. Zpravodajský článek o účasti Clintona na pohřbu Dizzyho Gillespieho může být zařazen do kategorie Politika, nebo do kategorie Jazz, nebo do obou kategorií, nebo dokonce do žádné z nich, v závislosti na subjektivním úsudku experta.

6.1.1 Klasifikace do jedné nebo více tříd

V závislosti na aplikaci mohou být na úlohu klasifikace textu uvalena různá omezení. Například můžeme požadovat, aby pro dané celé číslo k, označující počet klasifikačních tříd $\mathcal{C}$, bylo každému dokumentu $d_j \in \mathcal{D}$ přiřazeno přesně $l \leq k$, nebo $l \geq k$ tříd $\mathcal{C}$. Případ, kdy každému $d_j \in \mathcal{D}$ musí být přiřazena přesně jedna třída, se často označuje jako klasifikace do jedné třídy (angl. single-label) nebo s nepřekrývajícím se zařazením. Případ, kdy lze témuž $d_j \in \mathcal{D}$ přiřadit libovolný počet kategorií od 0 do k, se nazývá klasifikace do více tříd (angl. multi-label), neboli s překrýváním. Speciálním případem klasifikace do jedné třídy je binární klasifikace textu, kde každé $d_j \in \mathcal{D}$ musí být přiřazeno buď kategorii c_i, nebo jejímu doplňku $\bar{c}_i$. Z teoretického hlediska je binární případ (tedy i do jedné třídy) obecnější než klasifikace do více tříd, protože algoritmus pro binární klasifikaci lze použít i pro klasifikaci do více tříd. Stačí pouze transformovat problém klasifikace do více tříd pod množinou $\{c1, \ldots, k\}$ na k nezávislých problémů binární klasifikace pod $\{c_i, \bar{c}_i\}$, pro $i = 1, \ldots, k$. To však vyžaduje, aby na sobě byly jednotlivé třídy stochasticky nezávislé, tj. aby pro libovolné c', c'' hodnota $f(d_j, c')$ nezávisela na hodnotě $f(d_j, c'')$ a naopak. Obráceně to neplatí. Algoritmus pro klasifikaci do více tříd nelze použít ani pro binární, ani pro klasifikaci do jedné třídy. Ve skutečnosti, je-li dán dokument d_j ke klasifikaci, může klasifikátor přiřadit d_j $l > 1$ tříd a nemusí být zřejmé, jak z nich vybrat tu „nejvhodnější“, nebo klasifikátor nemusí přiřadit d_j vůbec žádnou třídu a nemusí být zřejmé, jak z $\mathcal{C}$ vybrat tu „nejméně nevhodnou“ [Sebastiani, 2001].

6.2 Předzpracování textu

Většina textových dokumentů pro použití v úloze strojového učení obvykle nevypadají tak, jak je potřeba, aby vypadaly. Standardně je takové dokumenty třeba nejprve upravit. Provést některé úpravy textu, aby byl text čistší, slovník menší nebo korpus vhodnější podle úlohy, kterou chceme řešit. Obecně je předzpracování textu část algoritmu, která může razantně navýšit úspěšnost použití dat při následném učení a klasifikaci, nicméně je důležité si uvědomit, že většina metod předzpracování textu má nevratný vliv na data. Při odstranění některých částí zdrojového textu, či dokonce slov už tato nejsou k dispozici a je možné tak přijít o důležitou informaci. Předzpracování textu může tedy nejen zvýšit, ale i snížit výslednou úspěšnost.

6.2.1 Čištění textu

Čištění textu je zpravidla úloha závislá na datech, která jsou využívaná. Je-li text nebo jeho části nějakým způsobem formátovaný, např. obsahuje tagy nějakého značkovacího jazyka (HTML, XML), které nejsou pro následné rozpoznávání žádoucí, je možné je odstranit. Dále je možné se při čištění textu zaměřit na znaky nebo sekvence znaků, které nepředstavují slova. Ať už se může jednat o alfanumerické kódy, nepísmenné sekvence, které nenesou informaci, jen mají pro člověka např. grafický význam z hlediska oddělení dvou sekcí nebo se dokonce může jednat o řetězce, které určitou informaci nesou, ale při návrhu systému se expert rozhodne, že pro danou úlohu nejsou relevantní. Zde se může jednat, vždy v závislosti na daném klasifikačním problému, o detekovatelné řetězce, jako jsou e-mailové adresy, adresy webových stránek, čísla, datum nebo čas.

6.2.2 Tokenizace

Tokenizace je proces převodu textu na seznam tokenů, což mohou být slova, fráze, symboly nebo jiné prvky významné z hlediska řešeného problému. V úloze klasifikace textu jsou smysluplnými prvky obvykle slova. Každý jazyk má specifická gramatická pravidla, která určují, jak používat interpunkci při skloňování slov nebo spojování vět, proto by každý jazyk měl mít také specifická pravidla, jak rozpoznávat a oddělovat tokeny.

Obvyklý proces tokenizace ve většině jazyků spočívá v tom, že se všechny symboly, které nemohou být součástí vlastního slova (interpunkce, nepísmenné symboly, atd.), oddělí od kontextu pomocí symbolu mezery, poté se text rozdělí podle bílých znaků (mezery, tabulátory, nové řádky) a každý oddělený kousek textu se zkontroluje podle sady regulárních výrazů závislých na jazyce, aby se ověřilo, že je daný kus textu smysluplný token [Lehečka, 2015].

6.2.3 Case folding a truecasing

V mnoha jazycích je zvykem psát první písmeno ve větě velkým písmenem. Jsou také další jazyková pravidla, která vedou k výskytu stejného slova s různou velikostí písmen, např. v němčině jsou psána všechna podstatná jména s velkým písmenem. Ačkoliv pro člověka nejsou slova: auto, Auto a AUTO významově rozdílná, pro počítač jsou to 3 různé řetězce textu. Nejjednodušší způsob, jak sjednotit různé velikosti písmen jednotlivých slov, je provést tzv. „case-folding“, označovaný také jako „lowercasing“, tj. převést každé písmeno v textu na jeho malou variantu. Přestože tato metoda může spolu

s korektním převodem různých textových řetězců na stejná slova převést také taková, která by měla být oddělena (např. vlastní jména a běžná slova jako třeba v angličtině „Bush“ a „bush“), uvádí se, že celkový dopad na úlohu klasifikace dokumentů je pozitivní [Uysal, Gunal, 2014].

Náročnější variantou je rozlišovat i slova s různou kapitalizací nebo dokonce opravovat špatně napsaný či nenapsaný text. K tomu slouží metoda „truecasing“, jejímž cílem je určit správnou kapitalizaci každého slova. Nástroje truecasing obvykle rozhodují o správné kapitalizaci na základě několika vlastností slova (např. pozice ve větě, frekvence varianty s malým písmenem v textu atd.) Truecasing je důležitou součástí mnoha úloh zpracování přirozeného jazyka, např. rozpoznávání pojmenovaných entit nebo strojového překladu, nicméně nejsou objektivní důkazy o pozitivním dopadu na úlohu klasifikace textu.

6.2.4 Stemming a lemmatizace

Metody stemming a lemmatizace jsou hlavními metodami procesu normalizace slov. Jsou vhodné pro použití pro jazyky, které skloňují nebo jinak ohýbají slova, mezi které patří např. čeština. Pomocí těchto dvou metod je možné všechny tyto tvary slova seskupit do jednoho tokenu. Stemming a lemmatizace se často směšují, ale výrazně se liší.

Stemming je proces převodu skloňované formy slova na jeho kmen, což je společná část všech skloňovaných variant slova. Stemming se obvykle provádí tak, že se z konců slov odstraní známé odvozovací afixy, neboli koncovky slov, v naději, že zůstane pouze kmen. Jedním z nejoblíbenějších algoritmů široce používaných pro angličtinu je Porterův algoritmus [Porter, 1980]. Naproti tomu **lemmatizace** je sofistikovanější proces, jehož cílem je převést skloňovaná slova na jejich základní lexikální formy (lemmata) pomocí slovníku a morfologické analýzy. Obě metody normalizace slov jsou závislé na jazyce textu a na stupni skloňování. Použití normalizace slov vede ve většině případů ke zmenšení rozsahu použitého slovníku. Čím více je jazyk skloňován, tím více se slovník může zmenšit.

Použití těchto metod nezaručí vždy zlepšení a existuje několik článků, které popisují jak zlepšení, tak zhoršení pro konkrétní jazyk a použitý korpus. Nelze ani říct, jestli je pro některý jazyk tato normalizace lepší, než pro jiný, protože existují výzkumy, které např. pro češtinu ukazují na zlepšení, ale jiné, ve kterých vyšly výsledky hůře [Skorkovská, 2012]; [Toman a kol., 2006]. Různé výsledky jednotlivých experimentů, a to i z hlediska zlepšení či zhoršení při použití normalizace slov vede k hypotéze, že u jazyků, které hodně skloňují, velmi záleží na daném korpusu dokumentů, a kolik obsahuje slov,

která po normalizaci ztratí svojí původní sémantickou informaci, která by mohla být důležitá pro následnou klasifikaci.

6.2.5 Odstranění nechtěných slov

Další metodou, která výrazně ovlivňuje výsledek, je vytvoření seznamu konkrétních slov, která budou odstraněna. Často se jedná o vysoce frekventovaná slova, která mají v korpusu obvykle jen malý lexikální význam. Jejich ignorováním se zmenší rozsah slovníku. Odstranění takových tematicky neutrálních slov z korpusu (nebo jejich prosté ignorování při práci s daty) je proces, který se nazývá odstranění stop slov (angl. stop word removal), neboli nechtěných slov. Univerzální seznam stop slov neexistuje a je nutné jej sestavit ručně. To platí pro každý jazyk zvlášť a často i pro konkrétní úlohu. Může se jednat např. o seznam všech funkčních slov daného jazyka nebo je možné jej částečně navrhnout automaticky tak, že se do seznamu stop slov zařadí všechna slova, která se vyskytují v dokumentech s frekvencí vyšší, než je určitá hodnota. K tomu může pomoci i výpočet idf (viz sekce 6.3.1). V některých úlohách může být rozumné zařadit do seznamu stop slov také veškerou interpunkci. Na rozdíl od jiných metod předzpracování textu, odstranění stop slov má na úlohu kategorizace textu pozitivní vliv, což bylo experimentálně potvrzeno [Uysal, Gunal, 2014]. Dokonce i automatický výběr stop slov na základě jejich frekvence výskytu v dokumentech může výrazně snížit chybu klasifikace ve srovnání s obecným seznamem stop slov [Echeverry-Correa a kol., 2015].

6.3 Reprezentace dokumentů

Zdrojová textová data jsou i po aplikaci metod předzpracování surového textu stále množinou textových řetězců, resp. tokenů. Naproti tomu algoritmy pro strojové učení očekávají vektorovou reprezentaci, a to ještě většinou potřebují, aby vektory byly stejně dlouhé i přes to, že každý dokument je jinak dlouhý a obsahuje jiný počet unikátních slov. Z tohoto důvodu je potřeba textové řetězce vhodně převést na číselné vektory o stejné délce, resp. dimenzi.

6.3.1 Bag-of-words a TF-IDF

Jednou z nejpoužívanějších metod pro reprezentaci vysokodimenzionálních dat, jako je text nebo také např. obrazová data (úlohy rozpoznávání obrázků) je metoda Bag-of-words. Aplikací tohoto přístupu je každý dokument reprezentován vektorem, jehož

hodnoty odpovídají vahám jednotlivých slov. Text je tedy doslova „sesypán“ do jednoho „pytle“ slov. Mezi hlavní výhody Bag-of-words patří jeho srozumitelnost, kdy je jasné, co který příznak znamená a často překvapivá přesnost.

Bag-of-words má však dvě hlavní omezení. Za prvé rozděluje termíny na slova, která je tvoří, např. „klasifikace textu“ rozděluje na slova „textu“ a „klasifikace“, za druhé považuje synonymní slova za nezávislé příznaky, např. „klasifikace“ a „kategorizace“ jsou považovány za dvě nezávislá slova bez sémantické asociace. Možným řešením těchto problémů je reprezentovat text pomocí pojmů, nikoliv jen jednotlivých slov. Tento přístup se pak nazývá *Bag-of-Concepts*. Hlavní výhodou přístupu Bag-of-Concepts je, že zachycuje a zachovává sémantiku a asociace mezi slovy vyskytujícími se v dokumentu a také jeho srozumitelnost. Nicméně pro jeho použití je nutná existence sémantické znalostní báze, jako je např. WordNet [Miller, 1995]. Další variantou adaptace Bag-of-words je použití n-gramů. *N-gramy* jsou sekvence po sobě jdoucích slov, které oproti jednotlivým slovům (které se dají považovat za 1-gramy neboli unigramy) zachycují i sémantiku N-slovných termínů. Nevýhodou N-gramů je rapidní zvýšení dimenze příznakového vektoru, protože každé slovo se ve slovníku vyskytuje několikrát, a to se svými předchůdci a následníky [Alahmadi a kol., 2013]. Nicméně vysoká dimensionalita příznakových vektorů je nevýhodou všech metod založených na principu Bag-of-words.

TF-IDF

TF-IDF je nejpoužívanější metodika k určení vah slov dokumentu v rámci použitého korpusu pro reprezentaci Bag-of-words. Jedná se o spojení četnosti slova v dokumentu, angl. **T**erm **F**requency a převrácené četnosti slova ve všech dokumentech, angl. **I**nverse **D**ocument **F**requency. Vychází z předpokladu, že termín t, který se v dokumentu d vyskytuje častěji, má pro dokument vyšší význam, než termín, který se vyskytuje méně často a zároveň termín, který se vyskytuje v mnoha dokumentech, není dobrým diskriminátorem a měl by mít menší váhu než takový, který se vyskytuje jen v několika dokumentech.

První část, TF, je v základu jednoduchá frekvence daného termínu v dokumentu, tedy kolikrát se v něm daný termín vyskytne:

$$tf_{td} = N(t, d). \tag{6.1}$$

V praxi se ale často používá některá z normalizovaných variant TF, a to buďto *kosinová normalizace* (6.2), která škálováním všech hodnot do rozsahu $\langle 0, 1 \rangle$ penalizuje, zahrnutím frekvencí všech slov dokumentu V, včetně opakování, dokumenty s termíny s vysokou

frekvencí

$$tf^c_{td} = \frac{tf_{td}}{\sqrt{tf^2_{t_1d} + tf^2_{t_2d}} + \cdots + tf^2_{t_Vd}} \tag{6.2}$$

nebo *sublineární škálování* (angl. sublinear tf scaling) (6.3), které škáluje nenulové výskyty na logaritmickou stupnici

$$tf^s_{td} = \begin{cases} 1 + \log tf_{td} & tf_{td} > 0 \\ 0 & jinak. \end{cases} \tag{6.3}$$

IDF, jakožto druhá část vztahu, přidává pohled na termíny z hlediska jejich výskytu napříč celým korpusem. Jedná se o logaritmus převrácené hodnoty počtu dokumentů, které obsahují daný termín $N(t)$ a počtu dokumentů v korpusu N:

$$idf_t = \log \frac{N}{N(t)}. \tag{6.4}$$

Tento vztah vysoce diskriminuje termíny, které jsou obsaženy ve všech dokumentech. Hodnota IDF takového termínu bude 0. Naopak termíny obsažené v málo dokumentech získají vysoké hodnocení.

Vhodným spojením vztahů *tf* a *idf* vznikne mocný a v klasifikaci textu často používaný nástroj na vektorizaci textových dokumentů.

$$tf\text{-}idf_{td} = tf_{td}(1 + idf_t) = tf_{td} + tf_{td} \cdot idf_t. \tag{6.5}$$

Vhodné spojení zde znamená, že standardně používaný výsledný vztah (6.5) není pouze jednoduchým vynásobením TF a IDF, protože termíny vyskytující se ve všech dokumentech, kde IDF $= 0$ by byly kompletně ignorovány, ale IDF je zvýšeno o 1, čímž k vynulování vztahu nedojde [Lehečka, 2015].

7

Vývoj vlastní metody učení s částečnou informací od učitele

Jak bylo naznačeno v kapitole 4, firmy dnes často mají historická data, která by mohla být využita k jejímu rozvoji, ke zlepšení služeb zákazníkům nebo ke snížení provozních nákladů. Pro tuto myšlenku jsem začal studovat tuto problematiku a dospěl k vývoji vlastního algoritmu učení s částečnou informací od učitele. Ten je popsán v této kapitole, a to včetně možného uplatnění, kterým by mohla být klasifikace příchozí e-mailové komunikace.

7.1 Vlastní metoda

Základní výhodou učení s částečnou informací od učitele je, že pro její aplikaci není potřeba velké množství anotovaných dat, která by sama o sobě vedla k úspěšnému klasifikování, ale jsou k dispozici testovací data, u kterých sice jejich zařazení není předem známé, ale je možné je použít k doplnění trénovacích dat a zlepšení úspěšnosti. Pro vývoj metody je vhodné mít anotovaný celý korpus, z něhož je část využita jako základní trénovací sada a další jako neanotovaná část, ve které jsou značky skryty a validační. To je vhodné pro měření úspěšnosti algoritmu, detekci chybných zařazení původně neanotovaných dat do pseudoanotovaných a na konečné měření úspěšnosti na validačních datech, která jsou po celou dobu procesu stálá.

7.1.1 Použitá data

Pro vývoj metody jsem zvolil všeobecně známá, veřejná a lehce dostupná data 20NewsGroups (http://qwone.com/~jason/20Newsgroups/). Jedná se o 3. verzi korpusu,

která je očištěna o duplicity, tedy o kolekci 18828 anglických textů rozdělených do 20 tříd. Třídy mají většinou více než 980 dokumentů, nejvíce 999, a některé z nich tvoří tématicky blízké skupiny.

Při představě reálné situace, kdy existuje velké množství neanotovaných dat, ze kterých je člověk schopen označkovat za krátký čas jen pár desítek, je nutné data pro použití v algoritmu rozdělit na části ve stejném duchu. Data jsou pro účely vývoje algoritmu rozdělena na tři části, a to na anotovaná data, neanotovaná data a validační data. Anotovaná část představuje malé množství „ručně označkovaných" dat. Druhou částí je množina neanotovaných dat, u které sice jejich správné zařazení známe, ale pro algoritmus je jejich označení skryto. Z této množiny, která je daleko obsáhlejší než první zmíněná, bude algoritmus vybírat vhodné adepty k automatické pseudoanotaci. Třetí část slouží pro validační účely algoritmu. Nezapadá již do analogie reálné situace, ale umožní nezávislé hodnocení úspěšnosti algoritmu tím, že poskytne jako neměnná množina dat komplexní pohled na vývoj úspěšnosti přes celý běh algoritmu.

Poměr objemu dat v jednotlivých částech: anotované, neanotované a validační, jsem zvolil 1:45:45, což odpovídá cca 10 dokumentům anotovaných dat pro každou třídu, na kterých algoritmus začíná a zbytek 50:50 pro neanotovanou a validační část. Pro rozumný start je nutné algoritmu poskytnout alespoň nějaká data. 10 dokumentů je tedy rozumně velká množina na to, aby se na nich klasifikátory mohly něco naučit, ale dostatečně malá, aby takový počet byl v reálném případě někdo schopný ručně označkovat.

7.1.2 Algoritmus

Vlastní algoritmus vychází z obalovacích metod učení s částečnou informací od učitele popsaných v části 5.1.1. Jedná se o iterativní algoritmus, který na začátku dostane několik označkovaných obrazů od každé třídy a množinu dat, kterou má za úkol díky znalosti původních anotovaných dat rozdělit a vybrat z nich vhodné kandidáty na převzetí do anotované, resp. pseudoanotované části. Vzhledem k tomu, že původní anotovaná data a algoritmem získaná pseudoanotovaná data algoritmus společně považuje za trénovací množinu, používám v následujícím textu název *pseudoanotovaná data*, jako označení pro obě zmíněné množiny spojené dohromady. Algoritmus má blízko k metodě kooperativního učení, popsané v kapitole 5.1.1.2, který využívá jednoho pohledu na data. Nicméně na stejná data se svým pohledem „dívá" několik různých klasifikátorů, jejichž společné klasifikace jsou použity k výběru vhodných neanotovaných dat pro přesun do pseudoanotovaných, díky kterým znají klasifikátory v další iteraci větší část vstupního prostoru. Tím, že data,

která klasifikátory využívají, jsou pro každý z nich stejná, se stále jedná o kooperativní učení s jedním pohledem, byť do úlohy každý klasifikátor přináší „svůj pohled“.

Celý algoritmus je napsaný v jazyku Python 3, který disponuje velkým množstvím volně použitelných balíků metod pro práci s daty a úlohy strojového učení. Mezi nejobsáhlejší, a v mé práci velmi využívané, jsou balíky scikitlearn, které představují jednoduchý a efektivní způsob pro vývoj aplikace strojového učení [Pedregosa a kol., 2011].

7.1.2.1 Načtení a příprava dat

Algoritmus po jeho spuštění načte zvolená data a rozdělí je ve zmíněném poměru 1:45:45 na požadované části. Každá část obsahuje informaci o samotných datech, zdrojovém souboru a třídě, do které patří. Pseudoanotovaná část navíc obsahuje „získanou“ hodnotu třídy, tedy hodnotu automatické pseudoanotace, kterou klasifikátory využívají pro svojí práci. Na počátku, kdy algoritmus dostane pro trénování anotovaná data, je tato pseudoanotace vyplněna jejich správnou známou třídou. Tím je zajištěna konzistence dat pro budoucí iterace. Pro načítání algoritmus používá funkci `sklearn.datasets.load_files`, která předpokládá konkrétní vstupní strukturu dat. Budou-li tuto strukturu splňovat, můžou být použita jakákoliv data. Tato struktura sestává ze složky korpusu, v níž jsou složky jednotlivých tříd, které obsahují soubory s daty. Každý soubor představuje jeden dokument.

Data jsou následně rozdělena pomocí funkce `sklearn.model_selection.train_test_split`, která je zároveň kromě rozdělení ještě zamíchá, aby nebyla v původním pořadí. Funkce dělí zvolená data vždy na dvě části podle zvoleného poměru. Nejprve jsou od celého korpusu oddělena anotovaná data. Následně proběhne rozdělení zbytku na neanotovanou a validační část.

7.1.2.2 Předzpracování a vektorizace dat

Další fází je převedení textových dat do použitelné podoby pro algoritmy strojového učení, tedy do příznakových vektorů. Pro vektorizaci využívám objektu `sklearn.feature_extraction.text.TfidfVectorizer`, který zastřešuje potřebné procedury pro převedení textu do Bag-of-words reprezentace s využitím TF-IDF popsaným v části 6.3.1. Konkrétně zastane tokenizaci, počítání slov a dokumentů, normalizaci a vytvoření výsledných příznakových vektorů.

Jednou z nejdůležitějších částí je tokenizace textu. Objekt `TfidfVectorizer` dovoluje přepsat defaultní tokenizační funkci vlastní funkcí, čehož jsem využil a nastavil tak vlastní

pravidla pro detekci tokenů, jejich úpravu a výběr. Slova neboli tokeny jsou v textu hledána pomocí regulárních výrazů. Standardní nastavení objektu hledá dvou a více-znakový alfanumerický řetězec a interpunkci vždy považuje za oddělovač. Takové nastavení může být vhodné pro základní věty, ale v případě, že se v textu objeví např. e-mailové adresy, odkazy nebo podobné specifické řetězce, tokenizace je zničí. Jelikož pro tvorbu Bag-of-words modelu používám unigramy, jsou zde termíny „slovo“ a „token“ ekvivalentní.

Po prozkoumání části korpusu a seznámení se s podobou surového textu jsem dospěl k regulárnímu výrazu 7.1, který zachovává slova s tečkou uprostřed (např. e-mailové adresy) a zároveň nebere jako slova čísla, řetězce čísel s méně než třemi písmeny nebo když se v textu objeví datum.

```
regex = r"(?!\b\d*\w\w?\d*\b)\b\w[\w\.@]+\b"
```

Kód 7.1: Regulární výraz pro tokenizaci

Regulárním výrazem získané tokeny jsou následně převedeny na jejich lemmatizované tvary. K lemmatizování tokenů využívám balík Stanza, což je kolekce nástrojů pro lingvistickou analýzu, která disponuje 100 modely pro 68 jazyků, včetně češtiny. Celkově balík Stanza umí kromě lemmatizace i tokenizaci, rozdělení textu po větách, nalezení víceslovných tokenů, označení větných členů nebo rozpoznávání jmenných entit. Jde tedy velmi obsáhlý balík s mnoha cennými metodami pro zpracování přirozeného jazyka [Qi a kol., 2020]. Pro tuto úlohu z nich využívám jen lemmatizační část. A to zejména proto, že Stanza používá složitější tokenizaci než potřebuji, čímž je výpočetně mnohem náročnější. Pro zachování nebo odstranění některých typů tokenů, čehož dosahuji tokenizací pomocí regulárního výrazu 7.1, by bylo potřeba výsledné Stanza tokeny ještě upravit a filtrovat.

Lemmatizované tokeny jsou dále filtrovány pomocí seznamu stop slov. Ten je, podobně jako proces lemmatizace, jazykově závislý. Tedy, pro každý jazyk je nutné mít seznam vlastní. Pro anglické texty, na kterých algoritmus testuji, jsem zvolil středně obsáhlý seznam, který sestává z 529 slov rozdělených do kategorií zobrazených v tabulce 7.1 níže.

Obsah seznamu stop slov ovlivňuje na jedné straně velikost výsledného slovníku, na druhé straně výkonnost algoritmu. Krátký seznam do 150 slov bude relativně rychlý, ale odstraní jen velmi omezený počet slov. Při obsáhlém seznamu nad 1200 slov je potřeba značný čas pro strojové porovnání všech slov zpracovávaných dokumentů a seznamu. Nicméně, čím víc slov seznam obsahuje, tím větší má vliv na kratší dokumenty. Tedy je větší pravděpodobnost, že dokument bude obsahovat slovo k odstranění [Hvitfeldt, Silge, 2021].

Podstatná jména	Přídavná jména	Zájmena	Číslovky	Slovesa	Příslovce	Předložky	Spojky	Částice	Citoslovce	Funkční slova	Ostatní
82	48	62	13	64	137	50	34	3	7	20	9

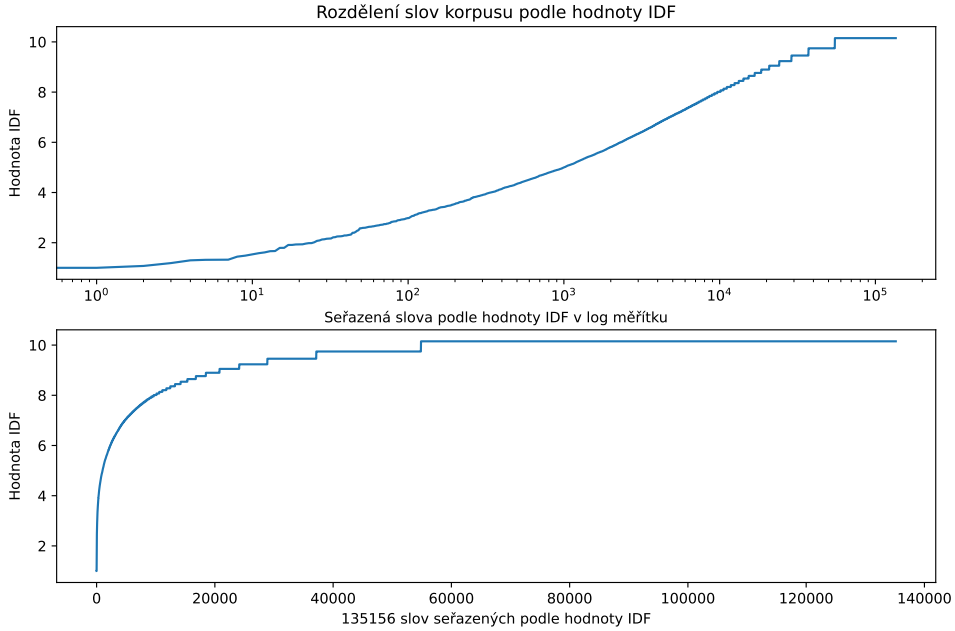
Tab. 7.1: Zastoupení slovních druhů v seznamu stop slov

Zmíněné procedury, tokenizace, lemmatizace a odstranění stop slov jsou vhodné nejen pro zmenšení šumu, tedy zavádějících informací, které se často v dokumentech objevují, ale také pro zmenšení slovníku neboli snížení dimenze příznakového vektoru, který při vektorizaci vzniká. Pro představu, na celém korpusu 20NewsGroups bylo při použití základní tokenizace objektu `TfidfVectorizer` nalezeno 153 330 tokenů. Vlastní regulární výraz snížil velikost „jen“ o necelé 4 000, ale zapojení lemmatizace přineslo pokles velikosti na 135 156, tedy o cca 18 000 tokenů. Odstranění stop slov může ze své podstaty snížit dimenzi nejvýše o svoji velikost. Výsledné snížení pro tento pokus je před vektorizací o 18 643 tokenů na 134 687.

Velikost příznakových vektorů značně ovlivňuje časovou náročnost trénování klasifikátorů a následné klasifikace. Nicméně po počátečním očištění dat a mírném snížení dimenze přichází mocnější nástroj pro odstranění tokenů z příznakového vektoru - odříznutí dle četnosti. K získání hodnot četností slov v rámci dokumentů a následné vektorizaci používám již zmíněný `TfidfVectorizer`. Ten po zmíněném předzpracování napočítá pro každý token jeho frekvenci výskytu pomocí TF-IDF popsaného v sekci 6.3.1 a vytvoří slovní reprezentaci Bag-of-words.

V obecném textu se objevují jak velmi frekventovaná slova, tak velmi málo frekventovaná slova. Těmi nejvíce frekventovanými bývají např. funkční slova jazyka nebo jiná, která často nenesou příliš informace. Také to naopak mohou být slova, která sice nositeli informace z hlediska významu jsou, ale tím, že se nachází ve většině dokumentů, nepřispívají k lepšímu rozlišení jednotlivých příznakových vektorů. Některá takto vysoce frekventovaná slova mohou být obsažena v seznamu stop slov, je ale očekávatelné, že značná část bude specifická pro použitý korpus. Jestliže není seznam stop slov vytvářen speciální technikou přímo pro daný korpus tak, aby přímo obsahoval nejfrekventovanější slova, je vhodné využít „odříznutí“ části příznaků dle jejich frekvence. Graf na obr. 7.1 ukazuje distribuci slov v korpusu a jejich výskyt v dokumentech podle hodnoty IDF. Vrchní graf v logaritmickém měřítku poskytuje detailnější pohled na slova, která se vyskytují téměř

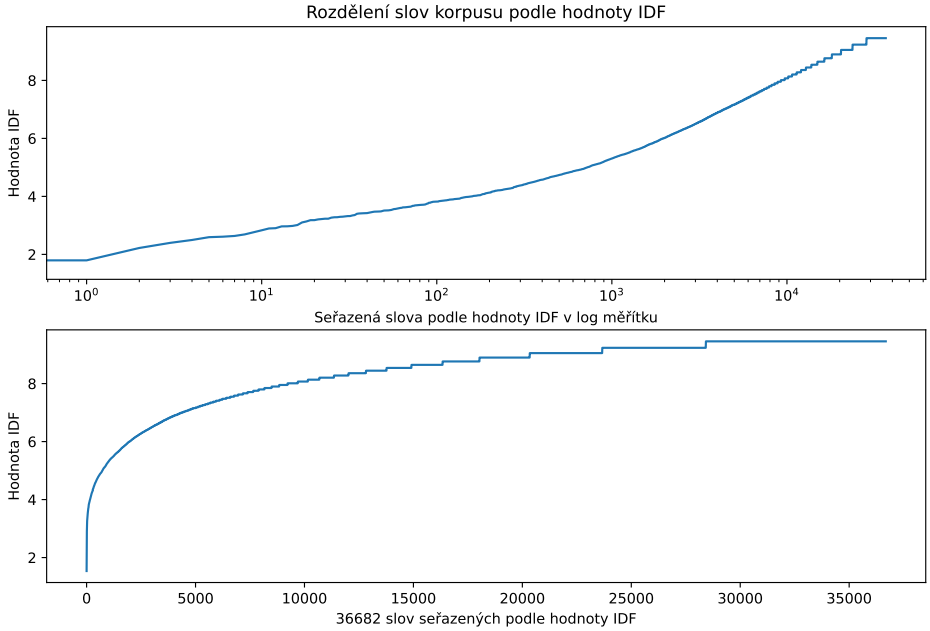
ve všech dokumentech, tedy mají nízkou inverzní hodnotu frekvence. Spodní graf lépe znázorňuje opačnou část distribuce, tedy slova, která se vyskytují v jednom dokumentu, ve dvou, atd. Každý schod v grafu přidá na četnost jeden dokument, ve kterém jsou daná slova obsažena.



Obr. 7.1: Dva pohledy na stejné rozdělení slov podle hodnoty inverzní frekvence dokumentů bez filtrace slov pomocí jejich IDF.

Na opačné straně se pak nacházejí slova, která se objevují v malém množství dokumentů a můžou tedy nést hodně informace z hlediska rozdělení obrazů, ale i tady jsou limity. Např. jestliže se určitý token nachází v trénovací části korpusu pouze jednou, je vskutku malá pravděpodobnost, že se bude nacházet v hodně dokumentech, které budou natrénované klasifikátory třídit. Každé odstranění části příznaků s sebou samozřejmě nese riziko ztráty informace, která mohla být užitečná pro klasifikaci. Nicméně v korpusu 20NewsGroups se nachází přes 80 000 tokenů, které se vyskytují pouze v jednom dokumentu, nepomohou tedy s klasifikací jiného textu, a to už je při odstranění těchto tokenů značné snížení dimenze. Pro porovnání je na obr. 7.2 znázorněna distribuce slov stejného korpusu s použitím filtrace slovníku pomocí hodnot IDF. Za povšimnutí stojí, že nastavením parametrů výskytu slova v minimálně 3 dokumentech a maximálně v 95 % dokumentů se slovník zmenšil téměř o 100 000 záznamů.

Zpracování surového textu na příznakové vektory je v porovnání s dalšími částmi úlohy časově velmi drahá operace. Např. vektorizovat korpus 20NewsGroups trvá na mém



Obr. 7.2: Dva pohledy na stejné rozdělení slov podle hodnoty inverzní frekvence dokumentů s použitou filtrací slov pomocí jejich IDF. Slovník obsahuje slova, která jsou alespoň ve 3 dokumentech a zároveň maximálně v 95 % dokumentů.

notebooku přibližně půl hodiny. Čas zpracování samozřejmě záleží na výkonu stroje, nicméně na běžném notebooku nižší střední třídy je takový korpus upočitatelný. Jakmile je vektorizace korpusu hotova, je vhodné si pro úsporu času zpracovaná data uložit, aby mohla být použita znovu. Pro tento účel využívám balíku `pickle`, který uloží zpracovaná data, včetně použitých objektů, po jejichž načtení může algoritmus pokračovat, jako by šlo o kontinuální běh.

7.1.2.3 Klasifikace a obohacení pseudoanotovaných dat

Získané příznakové vektory dále použiji jako vstupní parametry pro klasifikátory, které jsou v každé iteraci učeny pomocí pseudoanotované části dat a predikují zařazení do tříd pro neanotovanou a validační část. Jelikož jsem se rozhodl použít klasifikátory, které poskytuje balík `sklearn`, můžu je všechny v každé iteraci natrénovat pomocí příkazu `classifier.fit(X_train, y_train)`, kde objekt `classifier` ukazuje na konkrétní objekt trénovaného klasifikátoru, `fit()` je metoda, která natrénuje klasifikátor na množinu vstupních vektorů `X_train`, které patří do tříd definovaných stejně dlouhým polem značek `y_train`.

Pro získání predikcí neanotované a validační části dat je třeba na natrénovaném

klasifikátoru zavolat metodu `classifier.predict(X)`, kde parametr `X` je množina příznakových vektorů vždy jedné ze zmíněných částí dat. Metoda `predict()` vrátí pole značek tříd, které odpovídají jednotlivým vstupním příznakovým vektorům. Podle hodnot predikcí neanotovaných dat od jednotlivých klasifikátorů určuji, které obrazy budou vybrány a přesunuty do pseudoanotovaných dat. Rozhodnutí, které obrazy budou vybrány je závislé na velikosti shody predikce klasifikátorů a na jejich skóre.

První zmíněná podmínka je určena podílem klasifikátorů, kterých se nejvíce shodlo na jedné třídě. Tedy použiji-li 5 klasifikátorů, které predikují určitému obrazu třídy: $pred = [1, 1, 1, 4, 4]$, z počtu výskytů tříd 1 = 3x, 4 = 2x, je třída 1 zastoupena vícekrát, tedy obraz podle shody klasifikátorů patří do třídy 1 se shodou $\frac{3}{5}$.

Získání hodnoty skóre pro druhou podmínku by bylo nejjednodušší z pravděpodobností klasifikace, kdyby ji všechny klasifikátory poskytovaly. To se bohužel neděje, nicméně jsem pro algoritmus zvolil klasifikátory, které buď poskytují pravděpodobnost zařazení $p \in (0, 1\rangle$, nebo poskytují informaci o vzdálenosti obrazu od rozdělovací nadroviny d. Přímá konverze mezi vzdáleností d a pravděpodobností p není, ale pro získání aproximace pravděpodobnosti používám normalizaci 7.1.

$$d_{norm} = \frac{d - min(d)}{max(d) - min(d)} \tag{7.1}$$

Tato převede hodnotu vzdálenosti do rozsahu od 0 do 1, a je tak blíže pravděpodobnosti získané z jiných klasifikátorů. Po normalizaci tvořím skóre sumou získaných hodnot. To tedy může nabývat rozsahu od 0 do hodnoty počtu použitých klasifikátorů.

Všechny obrazy neanotovaných dat poté seřadím sestupně podle velikosti shody predikce klasifikátorů a následně podle jejich skóre. Toto seřazení je vhodné proto, že na vrcholu seřazeného seznamu obrazů se nachází takové, které nejvíce zapadají do natrénovaného modelu, níže pak takové, které jsou na základě pseudoanotovaných dat jen těžko zaklasifikovatelné. Ty se do pseudoanotovaných dat dostanou až jako poslední, čímž nekazí první iterace.

Při výběru obrazů, které se přidají do pseudoanotovaných dat je zachována rovnoměrnost naplnění jednotlivých tříd. Jinými slovy, nevybírají se striktně nejlepší kandidáti nehledě na třídu, do které je shoda klasifikátorů zařadila, ale pro každou třídu, která má být obohacena, je vybrán stejný počet obrazů.

V každé iteraci počítám hodnotu úspěšnosti klasifikace validačních dat. Ta by se měla s rostoucím počtem pseudoanotovaných dat, ze kterých se trénují klasifikátory, zvyšovat. Jestliže by se úspěšnost nezvyšovala, znamenalo by to, že nejsou splněny předpoklady pro učení s částečnou informací od učitele, což může mít různé důvody. Mezi nejčastější

patří nedostatečné předzpracování textu, kdy se do příznakových vektorů zanese příliš mnoho šumu nebo chybné označkování dat expertem.

Dalším krokem je přesun do další iterace, kdy se znovu vykonají procedury popsané v této části. Natrénují se klasifikátory na aktuálních pseudoanotovaných datech, díky nim se získají predikce zbylých neanotovaných dat a validační části korpusu a nejlepší kandidáti z množiny neanotovaných dat se pseudoanotují a přesunou do pseudoanotovaných dat. Takto algoritmus iteruje, dokud se nevyčerpají neanotovaná data nebo dokud neproběhne předem zvolený počet iterací.

7.2 Výsledky vlastní metody

Výše popsaný algoritmus si klade za cíl přinést zvýšení úspěšnosti klasifikace díky iterativnímu přidávání „vhodných“ kandidátů z množiny dostupných neanotovaných dat do pseudoanotovaných dat, ze které se klasifikátory trénují. Kromě již dříve popsaných parametrů, které je možné nebo nutné pro běh algoritmu zvolit, přichází samotný algoritmus ještě s dalšími. To může být jistou nevýhodou, protože ladění parametrů bývá obtížnou disciplínou u většiny algoritmů. Mezi hlavní otázky ohledně nastavení algoritmu patří:

- Jaké použít klasifikátory?
- Kolik klasifikátorů?
- Po kolika obrazech přidávat?

Odpovědi na tyto otázky poskytnou následující testy.

7.2.1 Porovnání klasifikátorů

K provedení experimentů jsem použil 6 druhů klasifikátorů, které jsou popsané v kapitole 3. Jelikož je metoda postavená na kooperaci několika klasifikátorů, rozdělil jsem je do dvou skupin po třech, abych porovnal, jaký vliv má volba klasifikátorů na její úspěšnost.

První skupinu tvoří multinomiální naivní Bayesův klasifikátor, klasifikace pomocí k-nejbližších sousedů a náhodný rozhodovací les. Všechny tři přímo pracují s pravděpodobností, kterou použité objekty balíku `scikit-learn` vrací spolu s výslednými predikcemi. Vstupní parametry experimentu jsou následující:

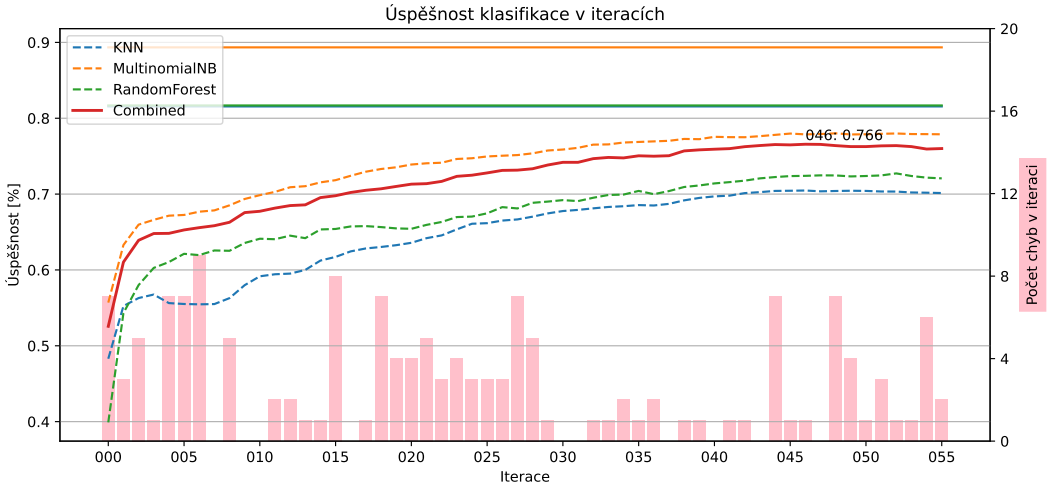
- Korpus dat 20NewsGroups obsahující 18828 dokumentů ve 20 třídách.

- Poměr anotovaných, neanotovaných a validačních dat 1:45:45, což vychází cca 10 obrazů na třídu v anotovaných datech.
- Při vektorizaci je použito lemmatizování tokenů, seznam stop slov a jsou odstraněny tokeny, které se vyskytují ve více než 95 % dokumentů, a které nejsou alespoň ve 3.
- Počet obrazů k přesunutí do pseudoanotovaných dat na třídu v každé iteraci = 10.
- Práh pro výběr kandidátů je nastaven na 1, tedy neaplikuje se.
- Maximální počet iterací = 60.

Parametry klasifikátorů jsou kromě standardně nastavených balíkem `scikit-learn` nastaveny takto: Multinomiální naivní Bayesův klasifikátor používá Lindstoneovo vyhlazování o hodnotě $\alpha = 0.01$, počet nejbližších sousedů pro kNN je $k = 5$ a náhodný rozhodovací les je bez změny standardních parametrů.

Pro všechny experimenty používám stejná data, která jsem rozdělil na neměnnou validační část a na 10 variant anotovaných a neanotovaných dat. Je tedy k dispozici 10 experimentů s různými počátečními podmínkami, stejnou výslednou množinou pseudoanotovaných dat a pevnými validačními daty, aby byly výsledky porovnatelné.

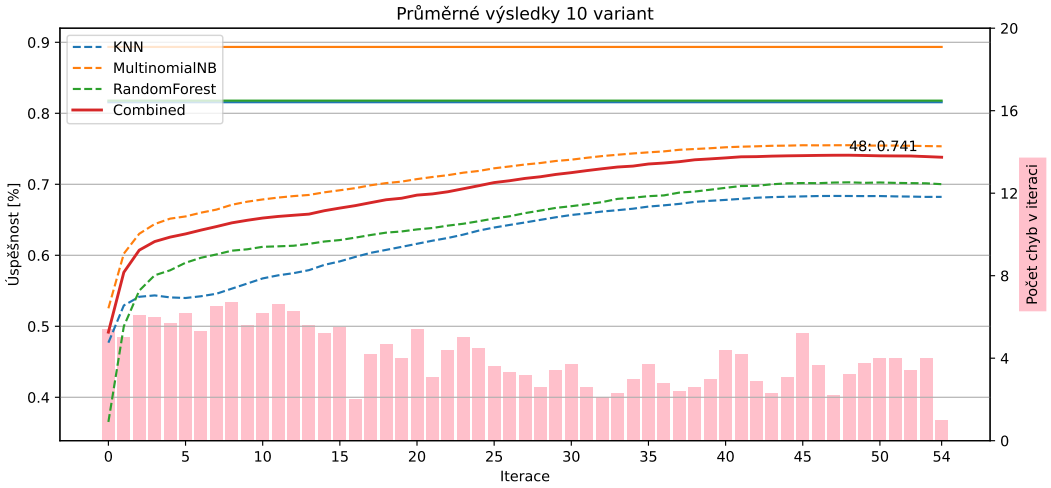
Následující grafy vždy zobrazují vývoj úspěšnosti klasifikování validačních dat v iteracích, které korespondují s hodnotami na levé svislé ose. Čárkovanou čarou jsou vykresleny úspěšnosti jednotlivých klasifikátorů, jejichž barvy jsou definovány v legendě, která nese jejich zkrácené názvy. Tučná čára pak vyznačuje vývoj úspěšnosti společné klasifikace, v legendě označen jako „Combined“. Plnými vodorovnými čarami, které odpovídají barvám čárkovaných čar úspěšností jednotlivých klasifikátorů jsou vyznačeny úspěšnosti, kterých by jednotlivé klasifikátory mohly teoreticky dosáhnout, když by algoritmus při vybírání obrazů a jejich označení nedělal chyby. Jinými slovy ukazují hodnotu získanou při natrénování klasifikátorů z anotované i neanotované části, při použití správných značek tříd. Spodní část, kde se nachází růžový sloupcový graf, ukazuje počet chyb, které v jednotlivých iteracích algoritmus udělal při vybírání a označkování obrazů z neanotovaných dat. Hodnoty korespondují s pravou svislou osou a jednotkou je počet kusů chyb. Z grafu na obr. 7.3 je vidět strmý vzestup úspěšnosti v prvních cca 3 iteracích, kdy se pseudoanotovaná data obohacují z původních 10 obrazů na 40. Následuje pomalejší nárůst s několika mírnými propady. Rychlý vzestup v prvních iteracích je očekávatelný, jelikož při prvním obohacení pseudoanotovaných dat je sada prakticky zdvojnásobena a také tím, že jsou do nich přesouvány „nejlepší“ kusy. Oproti ideální teoretické úspěšnosti, která je téměř 90 % pro multinomiální Bayesův klasifikátor a lehce nad 80 % pro zbylé dva klasifikátory využívající všechna správně anotovaná data, začínají klasifikátory použité v metodě poměrně nízko, s hodnotami úspěšnosti mezi 38 - 55 procenty. Celkový nárůst



Obr. 7.3: Nejlepší výsledek učení algoritmu s první sadou 3 klasifikátorů.

během iterací dosáhl v nejvyšším bodě 72 %, což je od počátečních 50 % 22% zlepšení. I tak je mezi teoretickým maximem a nejlepším výsledkem v 47. iteraci značný rozdíl, který je způsoben chybným vybíráním kandidátů pro přesun do pseudoanotovaných dat. Pro porovnání je na obr. A.7.1 graf nejhoršího průběhu běhu. Z důvodu lepšího přehledu a zvýšení kompaktnosti textu uvádím většinu grafů v příloze s odkazy v textu.

Jednotlivé běhy mohou být ovlivněny lepšími nebo horšími počátečními podmínkami. Protože jsou anotovaná a neanotovaná data vybírána náhodně, je pro ucelený pohled na úspěšnost vhodnější porovnávat průměr měření z několika běhů. Výsledek zprůměrování 10 experimentů je vidět na obr. 7.4. Už na první pohled je graf vyhlazenější,



Obr. 7.4: Průměr z 10 běhů učení algoritmu s první sadou 3 klasifikátorů.

tedy náhlá zvýšení úspěšnosti, či její propady jsou v průměru vykompenzována a z grafu je vidět, že s větším množstvím dat stoupá do určitého bodu i úspěšnost. Bod zlomu nastává

ve chvíli, kdy algoritmus udělá příliš mnoho chyb při přidávání obrazů do pseudoanotovaných dat. Graf potvrzuje rychlý nárůst úspěšnosti v prvních 3 iteracích a následný mírnější, který postupuje až ke 40. iteraci. Dále už je nárůst úspěšnosti pomalý, či dokonce klesá.

Druhou skupinou klasifikátorů, se kterými je první porovnána, tvoří: Ridge klasifikátor, Linear Support Vector klasifikátor a SVM s přístupem Stochastic Gradient Descent. Jejich parametry jsou kromě standardně nastavených balíkem `scikit-learn` nastaveny takto: Ridge klasifikátor používá gradientní metodu výpočtu vhodnou pro řídká data s tolerancí 1×10^{-2}, Linear Support Vector klasifikátor je nastaven na euklidovskou normu, toleranci 1×10^{-3} a jelikož se dá očekávat více příznaků, než obrazů, řeší primární optimalizační problém. Poslední použitý klasifikátor, SVM s přístupem Stochastic Gradient Descent používá pro penalizaci `Elastic Net` podpořenou penalizační konstantou $\alpha = 1 \times 10^{-3}$ s maximálním počtem průchodů přes trénovací množinu $max\_iter = 50$.

Tyto tři klasifikátory, jejichž průměrné výsledky z 10 běhů jsou zobrazeny na obr. A.7.2. Z porovnání grafů průměrných výsledků první a druhé sady klasifikátorů je vidět, že ačkoliv je druhá sada úspěšnější, jak v teoretickém maximu, tak na začátku běhu iterativní metody, tak nedosáhla vyšší společné úspěšnosti po vlastním označkování neanotovaných dat. To je způsobeno větší mírou nesouhlasu zvolených klasifikátorů.

Pro odpověď na otázku, jestli má vyšší teoretické maximum úspěšnosti klasifikace jednotlivých klasifikátorů pozitivní vliv na výsledek společného značkování neanotovaných dat přidávám ještě třetí sadu klasifikátorů, složenou z Linear Support Vector klasifikátoru, multinomiálního Bayesova klasifikátoru a Ridge klasifikátoru se stejným nastavením, jako bylo použito v miulých sadách. Z grafu na obr. A.7.3 je vidět, že tato hypotéza může být pravdivá, protože průměrný výsledek společné klasifikace je o 0.7 % vyšší, než u první sady. Znovu se tu opakuje stav, kdy má multinomiální Bayesův klasifikátor lepší individuální výsledky díky společnému značkování, než jsou výsledky společné klasifikace. Z toho vyplývá, že výsledným klasifikátorem, který z této iterativní metody vzejde nemusí být nutně spojení klasifikátorů použitých pro značkování, ale může být zřejmě některých případech použit jeden z nich, či dokonce jiný.

7.2.2 Porovnání počtu klasifikátorů

Z popisu kooperativního učení v sekci 5.1.1.2 plyne, že čím víc pohledů na data metoda má, tím lepší vykazuje výsledky. V mé iterativní metodě značkování neanotovaných dat jsou tyto pohledy zprostředkovány pomocí více různých klasifikátorů. Je taková různorodost pro úspěšnost metody dostatečná? Jestli ano, měl by být algoritmus, který

použije více klasifikátorů, úspěšnější, než ten, který jich použije méně. V předešlých sekcí výsledků již byly ukázány výsledky, jakých dosahuje algoritmus se třemi klasifikátory. Pro porovnání a odpověď na vznesenou hypotézu je možné vidět na obr. A.7.4 graf průměrných výsledků z 10 běhů algoritmu, který použil všech šest výše zmíněných klasifikátorů. Jestliže nejlepší výsledky, kterých dosáhly tři klasifikátory byly kolem 75 %, tak výsledek 77.7 % při použití šesti klasifikátorů podporuje hypotézu, že více pohledů přináší lepší výsledky.

7.2.3 Porovnání rychlostí rozšiřování trénovací množiny

Doposud bylo v každé iteraci označkováno 10 obrazů z množiny neanotovaných dat a přesunuto do pseudoanotovaných k následnému trénování klasifikátorů. Tato „rychlost“ značkování nemusí být nutně vhodnou volbou. Proto následující experimenty ukážou, jak důležitá je volba tohoto parametru.

Grafu na obrázcích A.7.6, A.7.3, A.7.7 a A.7.8 zobrazují průměrné výsledky z 10 běhů 3 klasifikátorů třetí sady s přidáváním 5, 10, 15, resp. 20 obrazů na třídu za iteraci. Největší rozdíl je v rychlosti, kterou algoritmus vyčerpal neanotovaná data, resp. poštu iterací, které provedl, než dospěl k výsledku. Protože byl pro experimenty nastaven maximální počet iterací 60, experiment s přidáváním po 5 obrazech na třídu nedoběhl až do vyčerpání dostupných dat. I z této počáteční části je v porovnání s ostatními variantami vidět, že výsledky úspěšnosti klasifikace se příliš nemění. Chybovost označkování v jednotlivých iteracích roste s více označkovanými obrazy, ale zdá se, že tato skutečnost nemá velký vliv na výsledky. Pro doplnění je ještě na obr. A.7.5 varianta běhu šesti klasifikátorů s rychlostí značkování 20 obrazů na třídu v iteraci, který jako varianty se třemi klasifikátory dopadl velmi podobně, a to s nejvyšší úspěšností 78 %.

Protože se výsledky při nastavení přesouvání od 10 do 20 obrazů příliš neměnily, přidávám pro porovnání ještě běhy s hodnotou parametru počtu nových označkovaných obrazů v iteraci 50, 100 a 200 pro experiment se třemi klasifikátory, jejichž výsledky jsou k vidění na obrázcích A.7.9, A.7.10 a A.7.11. Úspěšnost běhu s hodnotou parametru 50 jsou stále srovnatelné s předchozími. Až při 100 přidaných obrazech je chybovost algoritmu tak vysoká, že klesá výsledná úspěšnost, byť zatím jen trochu. Pokles dále potvrzuje výsledek 68 % z běhu s 200 nově označkovanými obrazy během jedné iterace. Nutno poznamenat, že ve všech případech byly klasifikátory na začátku trénovány pouze z 10 obrazů.

7.3 Možnosti praktického využití použité metody

Správa e-mailové korespondence je jednou z nejčastějších činností, kterou řeší snad každá firma a vynakládá na ni nemalé časové a finanční prostředky. Při představě větší firmy, které chodí denně třeba i 1500 e-mailů, jejichž průměrný čas zpracování je 5 minut, jde o čas, který je roven cca 15 lidem, kteří tyto e-maily zpracovávají. Budou-li průměrné náklady firmy na jednoho z nich 30 000 Kč, stojí takovou firmu vyřízení e-mailové korespondence minimálně 450 000 Kč měsíčně. Dále je nutné brát v úvahu, že každý zaměstnanec, který zpracovává e-maily z obecné schránky firmy, nemusí být proškolený na zpracování jakéhokoliv požadavku. Člověk tedy může přečíst několik e-mailů, než narazí na ten, který umí nebo má oprávnění zpracovat. I kdyby již přečtené e-maily, o kterých usoudí, že není správným člověkem pro jejich zpracování, poslal do dílčí schránky správnému zpracovateli, jeden e-mail je stále čten dvěma zaměstnanci, kteří na něm oba tráví stejný čas pouze čtením.

Množina e-mailů, která do firmy chodí, není složena jen z takových, které firmě přinesou další byznys, tedy vydělají peníze. Zákazníci často poptávají informace o stávajících produktech nebo nastavení smluv, které se jich přímo týká, tedy už je mají. Dalšími častými tématy mohou být např. dotazy týkající se plateb, kdy zákazník buď neví, co má kam zaplatit, poslal špatnou platbu, nemá na zaplacení a žádá jinou variantu, např. splátkový kalendář nebo žádá ukončení služby a odstoupení od smlouvy. Takové e-maily firmě moc výdělku nepřinesou. Mezi jednu z největších kategorií z hlediska objemu příchozích e-mailů ale patří e-maily, které nepotřebují další akci. Kromě spamů chodí spousta automatických odpovědí, oznámení o sjednané schůzce, které už stejně mají obě strany v kalendáři, nebo děkovné e-maily bez dalšího požadavku, které sice jistě vyjádřením spokojenosti potěší, ale kdo ví, jestli i ty, kteří platí nemalé peníze za to, že jimi jeho zaměstnanci tráví čas namísto výdělečnějších činností, které by mohli vytvářet.

Jestliže je cílem firmy snižovat provozní náklady, e-mailová korespondence často patří mezi kategorie, na kterých je možné použitím algoritmů strojového učení ušetřit nemalé sumy. Jestliže má firma historická data e-mailové komunikace, mohou taková data posloužit pro vytvoření automatického agenta, který by zaměstnancům, kteří zpracovávají některé typy e-mailových požadavků, zprávy roztřídil. Takový agent může být program, či skupina programů, která má přístup do firemní schránky, ze které vybírá příchozí e-maily, posílá je do fronty klasifikačnímu algoritmu, který navrhne správnou třídu pro jejich zařazení, kde může být další jeden nebo více programů, které vkládají roztříděné e-maily do již správných schránek lidem, kteří tak nemusí trávit čas čtením pro ně

nezpracovatelného e-mailu, nebo může e-mail, který nepotřebuje další interakci uživatele, rovnou vyřídit. Vyřídí ho např. tím, že ho v systému označí jako vyřízený, čímž agent firmě aktivně šetří zaměstnanecký čas, a tedy i peníze, protože dotčený e-mail nemusel nikdo ani jednou přečíst.

7.3.1 E-mail vs. obecný dokument

E-mail je obecně nestrukturovaný textový dokument, který nese oproti obecným textovým dokumentům zmíněným v sekci 6.1 další informace. Už základní zpráva se skládá ze dvou částí, a to z *předmětu zprávy*, který často nese stručnou informaci o obsahu, a z *těla*. Tělo e-mailu je většinou hlavním nositelem informace. Dále e-mail obsahuje metadata, jako je adresa odesílatele, adresa příjemce, datum odeslání, případně další příjemci kopie atd. Tato metadata je někdy možné využít pro částečné třídění e-mailů, a to dokonce i bez toho, aby musely být použity složité algoritmy. Takto snadná řešení však bývají použitelná pouze ve speciálních případech, kdy do schránky přicházejí zpravidla tytéž e-maily od často se opakujících odesílatelů, což se velmi blíží situacím, kdy schránce stačí nastavit několik ručních třídicích pravidel, a o jakémkoliv strojovém učení nemůže být řeč. Zpravidla jsou v náročnějších úlohách metadata spíše jakýmsi doplňkem, který může pomoct hlavní části - klasifikace těla a předmětu e-mailu.

Oproti obecným dokumentům jsou tedy v e-mailu dvě nosná pole, tělo a předmět, z nich ani jedno není povinné. Může se tedy stát, že přijde e-mail s textem v jeho těle, ale s prázdným předmětem, jiný s textem v předmětu, ale bez textu v těle nebo dokonce s prázdným jak tělem, tak předmětem, když např. zákazník posílá dokumenty, na čemž se domluvil s obsluhou telefonicky. Další rozdíl přinášejí e-mailové konverzace. Jestliže si dvě strany několikrát odpoví na původní e-mail, stane se, že tělo e-mailu se kromě textů jednotlivých zpráv, které můžou, ale taky nemusí nést informaci o celkovém tématu konverzace, zaplní metadaty, jako jsou hlavičky přeposlaných zpráv a odpovědí.

Celkově je e-mailová komunikace náročnějším typem dokumentu pro úlohy strojového učení, a to nejen pro větší počet datových částí, ale zejména pro část předzpracování. Aby mohlo být např. klasifikování e-mailů do kategorií podle témat úspěšné, musí být každý e-mail očištěn od nepotřebných a informaci nenesoucích částí, tedy musí dojít k pořádnému předzpracování surových e-mailů. Jestliže k takovému předzpracování dojde, je možné pak s e-maily zacházet velmi podobně, jako s jinými obecnými dokumenty. V takovém případě by mělo být možné použít iterační metodu značkování neanotovaných dokumentů, popsanou v této práci, i na e-mailové zprávy.

8

Závěr

Cílem této práce bylo shrnout poznatky učení s částečnou informací od učitele pro úlohu klasifikace dokumentů jako základ pro budoucí aplikaci takového přístupu v reálné úloze klasifikace korporátní e-mailové komunikace. Tyto poznatky vedly k návrhu vlastní metody iterativního značkování neanotovaných dat, které mohou být díky získaným označením zapojené do procesu učení.

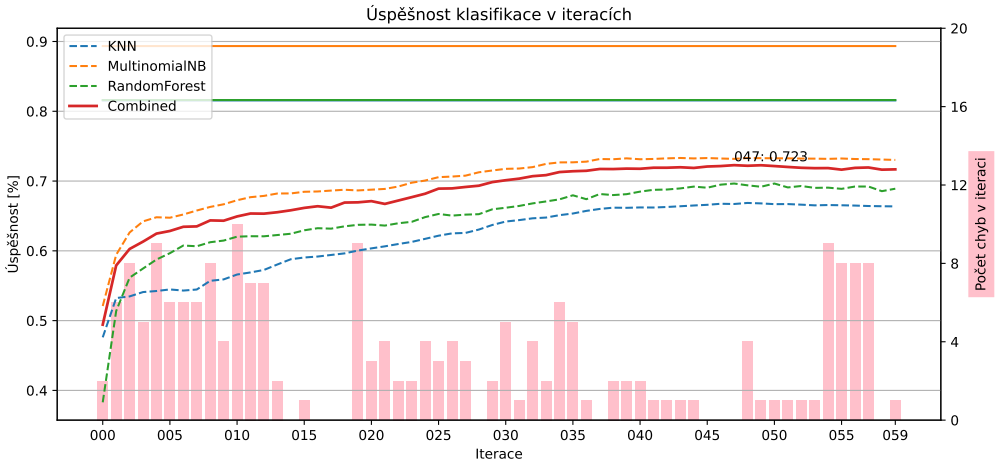
Metody učení s částečnou informací od učitele jsou mocné přístupy, které vyplňují mezeru mezi učením bez učitele a učením s učitelem. Úloh, kde by bylo vhodné nasadit algoritmy strojového učení, je mnoho. Nicméně většina z nich neposkytuje dostatečné množství kvalitně anotovaných dat pro přístupy učení s učitelem. Pro takové případy jsou metody učení s částečnou informací od učitele vhodné.

Jak bylo ukázáno v této práci, i relativně jednoduchá metoda založená na obalovacím přístupu kooperativního učení může zvýšit výkonnost klasifikátorů, které jinak vycházejí z klasického učení s učitelem. Při počátečním malém počtu 10 označkovaných obrazů pro každou třídu je popsaná iterativní metoda schopna zvýšit úspěšnost klasifikace i o více než 15 %, oproti výsledkům klasifikátorů natrénovaných pouze na počátečních 10 obrazech pro každou třídu.

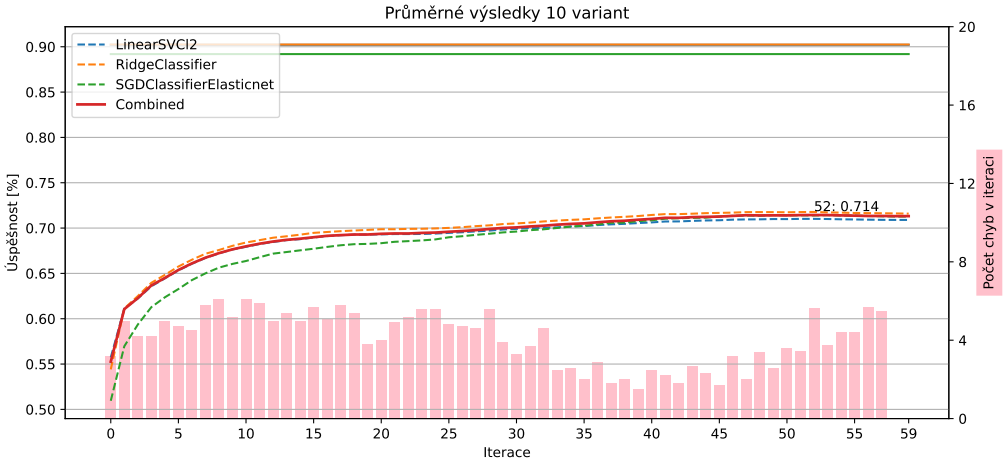
Výsledky v této práci popsané metody ukazují, že by s určitými úpravami, zejména adaptací na použitou datovou sadu, mohla být nasazena ve složitějších úlohách. Potvrzení této hypotézy by však potřebovalo další výzkum.

Příloha A

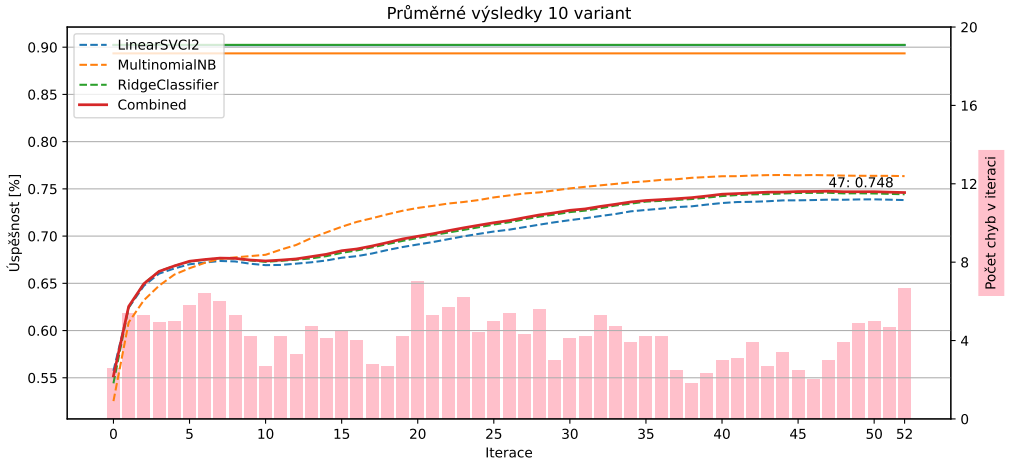
Grafy výsledků



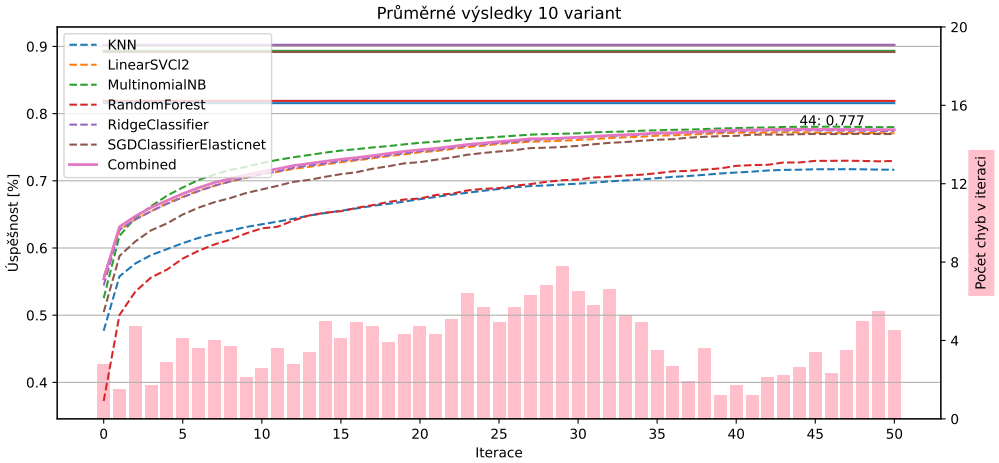
Obr. A.7.1: Nejhorší výsledek učení algoritmu s první sadou 3 klasifikátorů.



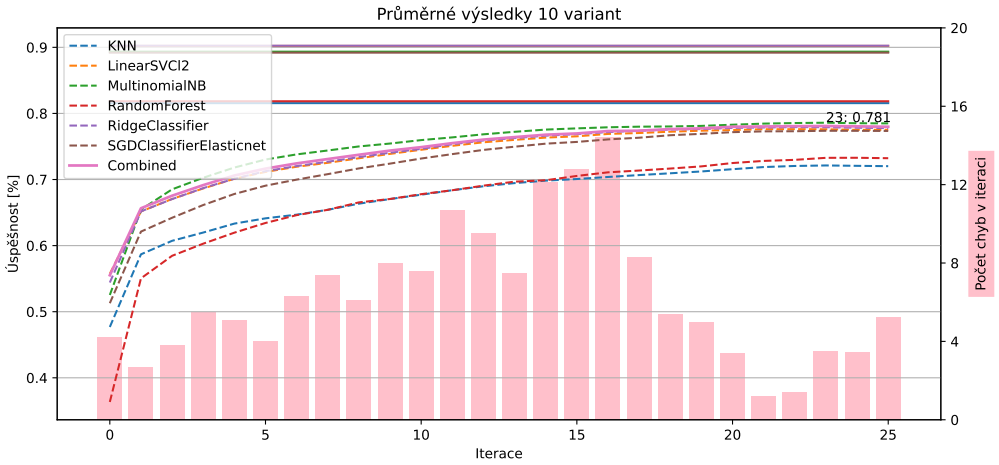
Obr. A.7.2: Průměr z 10 běhů učení algoritmu s druhou sadou 3 klasifikátorů.



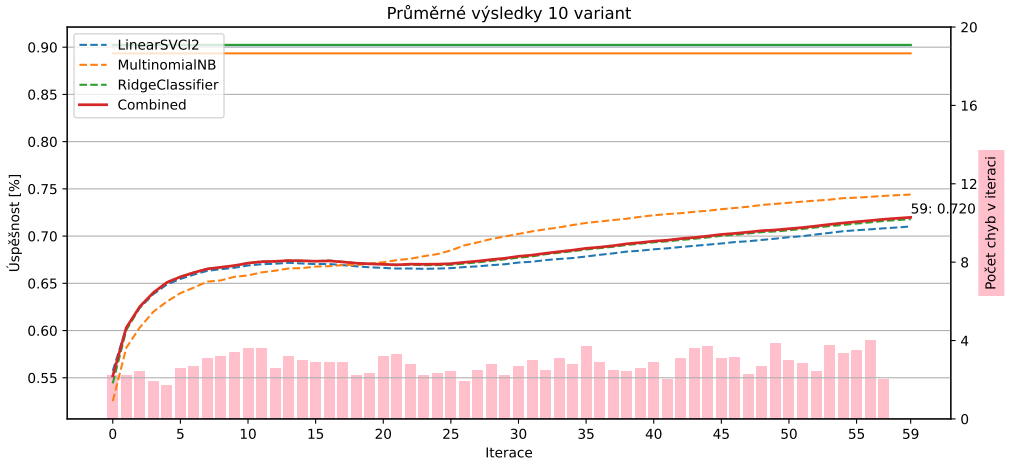
Obr. A.7.3: Průměr z 10 běhů učení algoritmu s třetí sadou 3 klasifikátorů.



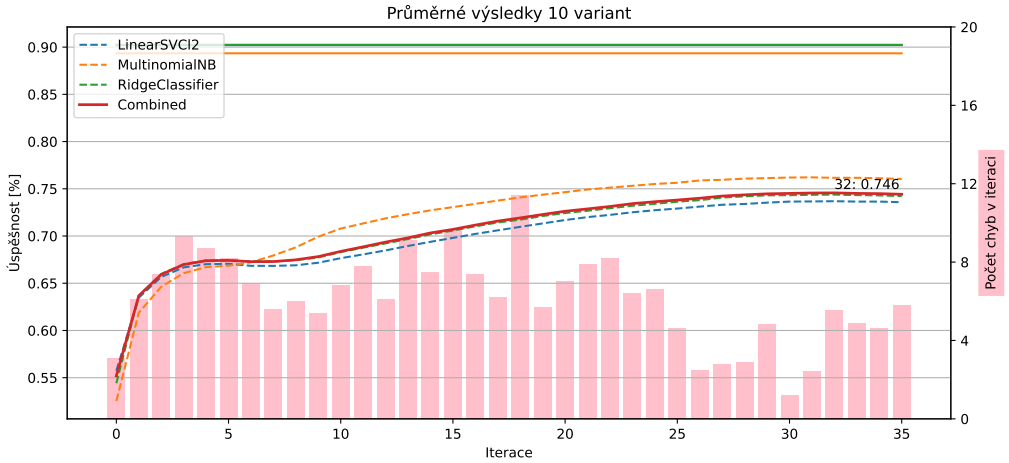
Obr. A.7.4: Průměr z 10 běhů učení algoritmu se sadou 6 klasifikátorů.



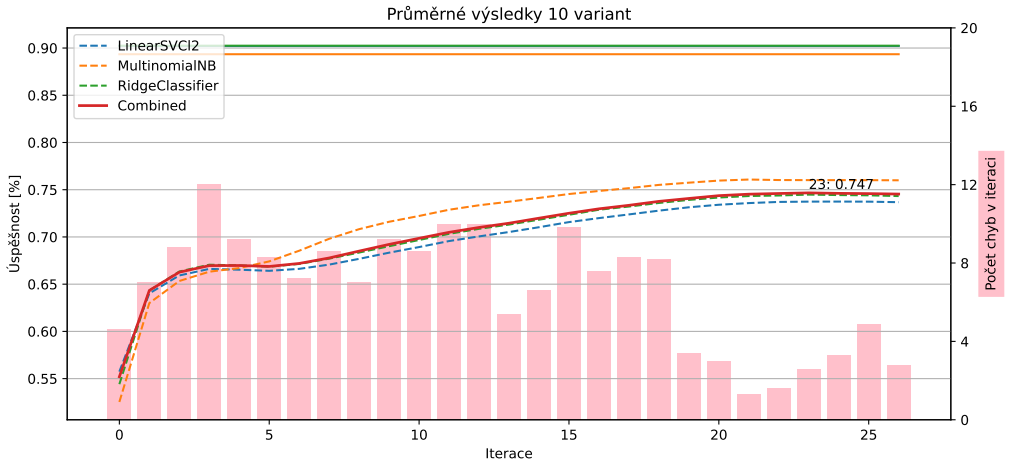
Obr. A.7.5: Výsledky sady 6 klasifikátorů, při přírůstku 20 obrazů na třídu za iteraci.



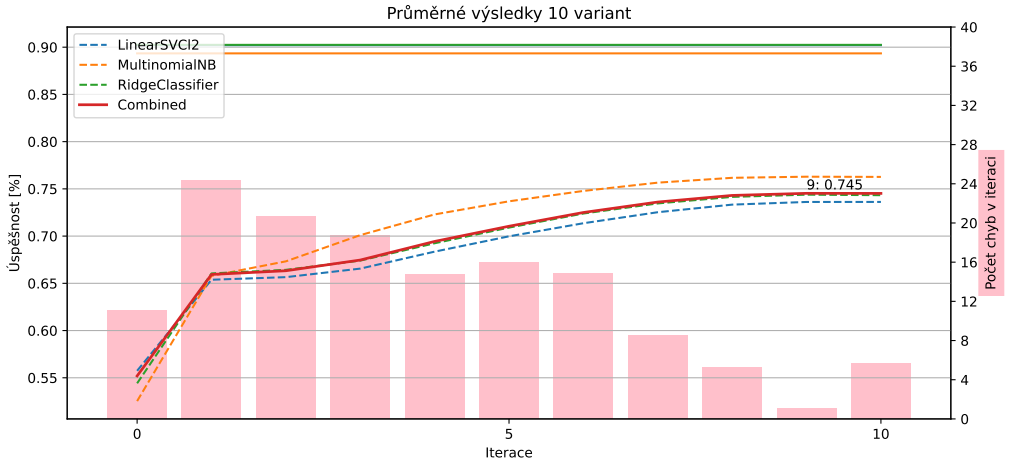
Obr. A.7.6: Výsledky třetí sady 3 klasifikátorů, při přírůstku 5 obrazů na třídu za iteraci.



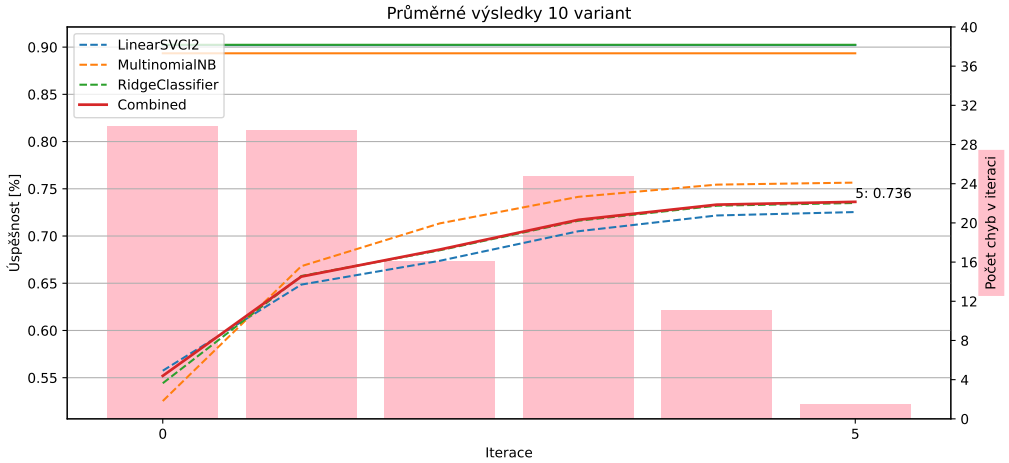
Obr. A.7.7: Výsledky třetí sady 3 klasifikátorů, při přírůstku 15 obrazů na třídu za iteraci.



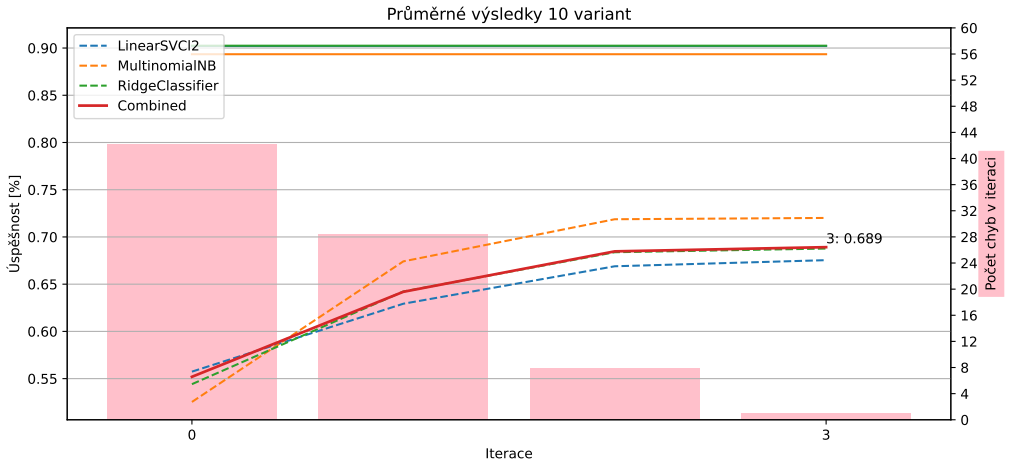
Obr. A.7.8: Výsledky třetí sady 3 klasifikátorů, při přírůstku 20 obrazů na třídu za iteraci.



Obr. A.7.9: Výsledky třetí sady 3 klasifikátorů, při přírůstku 50 obrazů na třídu za iteraci.



Obr. A.7.10: Výsledky třetí sady 3 klasifikátorů, při přírůstku 100 obrazů na třídu za iteraci.



Obr. A.7.11: Výsledky třetí sady 3 klasifikátorů, při přírůstku 200 obrazů na třídu za iteraci.

Literatura

[Abney, 2002] Abney, S. (2002). Bootstrapping. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 360–367, USA. Association for Computational Linguistics.

[Alahmadi a kol., 2013] Alahmadi, A., Joorabchi, A., and Mahdi, A. (2013). A new text representation scheme combining bag-of-words and bag-of-concepts approaches for automatic text classification. pages 108–113.

[Chapelle a kol., 2006a] Chapelle, O., Chi, M., and Zien, A. (2006a). A continuation method for semi-supervised svms. pages 185–192.

[Chapelle a kol., 2006b] Chapelle, O., Schölkopf, B., and Zien, A. (2006b). *Semi-Supervised Learning*.

[Chapelle a kol., 2009] Chapelle, O., Scholkopf, B., and A., Z. (2009). Review of semi-supervised learning by o. chapelle, b. schölkopf, and a. zien, eds. london, uk, mit press, 2006. *Neural Networks, IEEE Transactions on*, 20:542–542.

[Cleverdon, 1988] Cleverdon, C. (1988). Optimizing convenient online access to bibliographic databases. *Information Services and Use*, 4.

[Echeverry-Correa a kol., 2015] Echeverry-Correa, J., Ferreiros, J., Coucheiro-Limeres, A., Cordoba, R., and Montero, J. (2015). Topic identification techniques applied to dynamic language model adaptation for automatic speech recognition. *Expert Systems with Applications*, 42:101–112.

[Engelen, Hoos, 2020] Engelen, J. and Hoos, H. (2020). A survey on semi-supervised learning. *Machine Learning*, 109.

[Huang a kol., 2006] Huang, T., Kecman, V., and Kopriva, I. (2006). *Kernel Based Algorithms for Mining Huge Data Sets: Supervised, Semi-Supervised, and Unsupervised Learning*, volume 17.

[Hvitfeldt, Silge, 2021] Hvitfeldt, E. and Silge, J. (2021). *Supervised Machine Learning for Text Analysis in R*. Chapman & Hall/CRC Data Science Series. Taylor & Francis Limited.

[Kunchhal, 2020] Kunchhal, R. (2020). *The Mathematics Behind Support Vector Machine Algorithm (SVM)*. `https://www.analyticsvidhya.com/blog/2020/10/the-mathematics-behind-svm`. Accessed: 2021-08-20.

[Larkey, 1999] Larkey, L. (1999). A patent search and classification system. *Proceedings of the 4th ACM Conference on Digital Libraries*.

[Lehečka, 2015] Lehečka, J. (2015). *Adaptace jazykového modelu na téma v reálném čase: Online topic-based language model adaptation*.

[Miller, 1995] Miller, G. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, 38:39–.

[Oliver a kol., 2018] Oliver, A., Odena, A., Raffel, C., Cubuk, E., and Goodfellow, I. (2018). Realistic evaluation of deep semi-supervised learning algorithms.

[Pedregosa a kol., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Porter, 1980] Porter, M. (1980). An algorithm for suffix stripping. *Program: Electronic Library and Information Systems*, 14.

[Qi a kol., 2020] Qi, P., Zhang, Y., Zhang, Y., Bolton, J., and Manning, C. D. (2020). Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

[Rosenberg a kol., 2005] Rosenberg, C., Hebert, M., and Schneiderman, H. (2005). Semi-supervised self-training of object detection models. pages 29–36.

[Sebastiani, 2001] Sebastiani, F. (2001). Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1–47.

[Skorkovská, 2012] Skorkovská, L. (2012). Application of lemmatization and summarization methods in topic identification module for large scale language modeling data filtering. volume 7499.

[Tanha a kol., 2015] Tanha, J., Someren, M., bullet, S., and Afsarmanesh, H. (2015). Semi-supervised self-training for decision tree classifiers. *International Journal of Machine Learning and Cybernetics*, 24.

[Toman a kol., 2006] Toman, M., Tesar, R., and Jezek, K. (2006). Influence of word normalization on text classification.

[Triguero a kol., 2015] Triguero, I., García, S., and Herrera, F. (2015). Self-labeled techniques for semi-supervised learning: Taxonomy, software and empirical study. *Knowledge and Information Systems*, 42.

[Uysal, Gunal, 2014] Uysal, A. K. and Gunal, S. (2014). The impact of preprocessing on text classification. *Information Processing and Management*, 50:104 – 112.

[Wang, Zhang, 2008] Wang, F. and Zhang, C. (2008). Label propagation through linear neighborhoods. *IEEE Trans. Knowl. Data Eng.*, 20:55–67.

[Wang, Zhou, 2010] Wang, W. and Zhou, Z.-H. (2010). A new analysis of co-training. pages 1135–1142.

[Yarowsky, 1995] Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, ACL '95, page 189–196, USA. Association for Computational Linguistics.

[Zhou, 2012] Zhou, Z.-H. (2012). *Ensemble Methods: Foundations and Algorithms*, volume 14.

[Zhu, 2008] Zhu, X. (2008). Semi-supervised learning literature survey. *Comput Sci, University of Wisconsin-Madison*, 2.

[Zhu, Goldberg, 2009] Zhu, X. and Goldberg, A. (2009). *Introduction to Semi-Supervised Learning*, volume 3.