VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

KLASIFIKÁTOR EMAILOVÉ KOMUNIKACE

CLASSIFICATION OF EMAIL COMMUNICATION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. MAREK PIJÁK**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. IGOR SZŐKE, Ph.D.**
SUPERVISOR

BRNO 2018

Zadání diplomové práce/20270/2017/xpijak01

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2017/2018

Zadání diplomové práce

Řešitel: **Piják Marek, Bc.**
Obor: Počítačová grafika a multimédia
Téma: **Klasifikace emailové komunikace**
Classification of eMail Communication
Kategorie: Zpracování řeči a přirozeného jazyka

Pokyny:

1. Seznamte se se základy klasifikace témat ze textu a se základy machine learning.
2. Najděte vhodná data. Navrhněte jednoduchý klasifikátor témat. Otestujte úspěšnost klasifikace pomocí vhodně zvolené metriky.
3. Zdokonalte klasifikátor (např. více dat, lepší algoritmy, více tříd) a průběžně sledujte jeho úspěšnost.
4. Otestujte funkčnost klasifikátoru na reálném provozu. Diskutujte dosažené cíle a navrhněte směry dalšího vývoje.
5. Vyrobte A2 plakátek a cca 30 vteřinové video prezentující výsledky Vaší práce.

Literatura:

- Dle pokynů vedoucího
- http://www.wildml.com/2016/04/deep-learning-for-chatbots-part-1-introduction/

Při obhajobě semestrální části projektu je požadováno:

- Body 1 a 2 ze zadání.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci dřívějších projektů (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Szőke Igor, Ing., Ph.D.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 23. května 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Tato diplomová práce se zabývá vytvořením klasifikátoru, který bude schopen rozpoznat emailovou zprávu společnosti Topefekt.s.r.o a zařadit ji do korespondující klasifikační třídy. Tento projekt bude využívat řadu nejpoužívanějších klasifikačních metod včetně strojového učení. Jako součást této práce bude i ohodnocení úspěšnosti jednotlivých metod.

Abstract

This diploma's thesis is based around creating a classifier, which will be able to recognize an email communication received by Topefekt.s.r.o on daily basis and assigning it into classification class. This project will implement some of the most commonly used classification methods including machine learning. Thesis will also include evaluation comparing all used methods.

Klíčová slova

Klasifikace, Metoda, Email, Zpráva, Frekvence termínu, Redukce dimenzionality, Naïve Bayes, K-nejbližších sousedů, Support vector machine, SVM, F1 skóre, K-fold validace, Učení s učitelem, Confusion matice

Keywords

Classification, Method, Email, Message, Term Frequency, Reduction in dimensionality, Naïve Bayes, K-neighbors, Support vector machine, SVM, F1 score, K-fold validation, Supervised learning, Confusion matrix

Citace

PIJÁK, Marek. *Klasifikátor emailové komunikace*. Brno, 2018. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Igor Szőke, Ph.D.

Klasifikátor emailové komunikace

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Igora Szőkeho, Ph.D., Ing. Lubomíra Kozáka a Ing. Jakuba Hatoně.

.....
Marek Piják
20. května 2018

Poděkování

Tímto bych chtěl poděkovat zejména panu Igoru SzőkemuPh.D.,za aktivní přístup k vedení této práce a jeho užitečné rady. Dále bych chtěl poděkovat Ing. Lubomíru Kozákovi a Ing. Jakubovi Hatoňovi za příležitost tuto práci zrealizovat.

Obsah

Kapitola 1

Úvod

Tento dokument vznikl jako diplomová práce na Fakultě informačních technologií Vysokého učení technického v Brně. Cílem práce je vytvořit klasifikátor emailové komunikace společnosti TOPefekt.s.r.o, která působí v České republice a několika dalších zemích jako agregátor SMS zpráv. Klasifikátor bude poskytovat informace zaměstnancům, kteří se starají o zákaznickou podporu. Lidské zdroje jsou dnes nejdražší položkou ceny projektu, proto je dobré tyto zdroje ušetřit a vhodně je využít.

Je všeobecně známo, že převážná většina času v životním cyklu projektu se týká technické podpory, což obnáší náročné konverzace se zákazníky, kteří potřebují vyřešit svůj problém co nejdříve. Jeden z hlavních cílů firem je efektivita práce. Je tedy nežádoucí nasazovat velké množství lidských zdrojů na stále se opakující problémy, které mohou být řešeny automaticky. Daná problematika se komplikuje, pokud firma obchoduje se zahraničím. Dlouhá prodleva mezi odesláním dotazu, přeposlání zprávy skrze firemní hierarchii ke kompetentní osobě může být zdrojem komplikací pro zákazníky, kteří nemohou čekat. Proto je vhodné tento proces zautomatizovat a rozlišit jednotlivé zprávy na základě jejich obsahu a priorit.

Využití klasifikátoru pomůže urychlit řešení stále se opakujících dotazů a omezit čekání zákazníka na odpověď. Cílem je poskytnout vedoucím firmy informace o formě daného problému, což zjednoduší přesměrování daného problému na konkrétního technika a tím urychlit řešení problému.

1.1 Popis kapitol

Kapitola 2 popisuje důvody k tvorbě této práce a problémy které bude potřeba řešit. Kapitola 3 pojednává o tvaru získaných dat a jejich rozdělení do klasifikačních tříd a posléze jsou popsány i metody zpracování textu. Vybrané metody klasifikace a jejich popis obsahuje kapitola 4. Popis postupů a metod použitých při vytváření vstupních vektorů (matic) a klasifikačních modelů se nacházejí v kapitole 5. Kapitola 6 se zabývá experimenty s daty za účelem vylepšení přesnosti implementovaných modulů. Hodnocení jednotlivých kroků klasifikace a návrh budoucích vylepšení společně s testem reálného provozu najdeme v kapitole 7. Poslední kapitola 8 zhodnocuje výsledek práce.

Kapitola 2

Motivace

Před samotným započetím práce vyvstává otázka. Proč se tímto problémem vůbec zabývat? Jakým způsobem přispěje? Co je na tomto problému tak složité? V této kapitole si popíšeme důvody vzniku klasifikátoru emailové komunikace.

- **Zátěž** - Společnost TOPefekt.s.r.o je denně zaplavena emailovými zprávami, buďto legitimními, nebo zbytečnými. Programové urychlení eliminačního procesu, nebo případné okamžité přeposlání konkrétnímu člověku by uvolnilo spoustu lidského úsilí a tím i zvýšilo efektivitu práce.
- **Znalost angličtiny** - Převážná většina zákazníků této firmy pochází ze zahraničí. Je tedy součástí příjímacího řízení být schopný chápat, velice pokročilou angličtinu. Ať už se jedná o dlouhý komplikovaný text, nebo o rozbitou angličtinu, která je na překlad složitější. Automatizované rozpoznání kontextu by mohlo ulehčit práci a zpřístupnit pracovní pozici i pro méně kvalifikovaného zaměstnance.
- **Automatické přesměrování** - Za předpokladu, že bude klasifikace dostatečně přesná, bylo by možné rozšířit funkcionalitu na automatické přeposlání zprávy kvalifikovanému technikovi pro urychlení odpovědi zákazníkovi.
- **Časové úspory** - Firma TOPefekt.s.r.o si zakládá na pozitivním přístupu a vztahu se zákazníky. A zákazníci, kterým je rychle a relevantně odpovězeno vědí, že jsou jejich problémy brány vážně a je na nich pracováno.

2.1 Jaké jsou hlavní problémy

Klasifikace textu není v dnešní době nic nového, nicméně v jiných případech se jedná o klasifikaci legitimní pošty, nebo spamu. V jiných případech jde o klasifikaci na základě daného tématu. Nebo se může jednat o rozpoznání použitého jazyku. Náš dataset je poněkud komplikovanější.

- **Kombinace kontextů** - Zprávy nemají předem specifikovaný formát, což znamená, že zákazník může položit jeden stručný dotaz, nebo mnoho komplikovaných dotazů. Pokud je tedy možné z obsahu určit okruhy, kterých se dotaz týká, zkrátí se čas mezi obdržením zprávy a řešením problému.
- **Cizojazyčnost** - Jazyk zpráv se mění od zprávy ke zprávě, nebo dokonce v rámci jediné konverzace, což sebou přináší i různé znakové sady.

- **Kombinace různých jazyků** - Nejen, že zprávy se vyskytují ve spektru různých jazyků, ale objevují se i případy, kdy si zákazník není jistý, zdali zaměstnanci firmy znají daný jazyk. Výsledná zpráva tedy obsahuje zprávu ve dvou či více jazycích najednou.
- **Gramatika** - Když už se zákazník rozhodne kontaktovat firmu, musí učinit rozhodnutí, zdali sepíše zprávu ve svém rodném jazyce, nebo se pokusí domluvit angličtinou. Někteří lidé mají problémy s gramatikou svého rodného jazyka, co teprve když je to jejich druhý jazyk. Podíváme-li se na to z klasifikačního hlediska, správně zapsané slovo, je úplně jiné slovo, než špatně zapsané slovo.
- **Nerovnoměrnost zastoupení** - Spousta dotazů se během komunikace se zákazníky vyskytuje více než jiné. Není tedy jisté, jestli se určité zprávy vyskytují dostatečně často, aby bylo možné je rozpoznat.

Hlavní otázkou této práce je především, je vůbec možné tyto zprávy spolehlivě klasifikovat? A pokud ne, jaké kroky by se musely podstoupit, aby to možné bylo.

Kapitola 3

Analýza zpráv

Jelikož bude náš model trénován principem supervised learning, je nezbytné modelu poskytnout příklady zpráv patřících do konkrétních tříd klasifikace. Musíme tedy celý náš dataset přečíst, identifikovat jednotlivé třídy a následně je ručně roztřídit. Pokud bychom našemu klasifikátoru poskytli neroztříděná data, byl by pouze schopen říci, zdali tato zpráva pochází z emailových serverů dané firmy, ale nebyl by schopen říci nic o samotném obsahu.

Firma působí jako agregátor SMS zpráv, všechny emailové zprávy se tedy týkají technické podpory implementovaných SMS modulů. Příklady jednotlivých instancí klasifikačních tříd jsou uvedeny dále. Abychom nešířili soukromé informace zákazníků, byly všechny potenciálně zneužitelné informace nahrazeny řetězcem náhodné délky složeným ze symbolů *.

3.1 Support - nedoručené SMS/Simulace (zapnuta)

První třída Support se týká běžné zákaznické podpory obsahující požadavky typu, nebyla odeslána SMS zpráva, mám problém s danou částí systému, systém nefunguje, jak by měl. Bohužel příčina těchto závad může být zapříčiněna velkým množstvím problémů, není tudíž možné klasifikovat tyto zprávy dle konkrétního řešení, proto je vhodnější vytvořit jednu třídu pro všechny tyto zprávy.

```
Předmět: CART-SMS.COM - CONTACT
Datum: Wed, 4 Oct 2017 08:26:00 +0200
Od: ****** *** ***** <****************.com>
Komu: support@topefekt.com

Hello
There is a problem receiving messages from customers and also managers
Do not reach them
But when you enter the message report it appears that it has been sent
but They are received from the customer or administration Please work
on it as a problem quickly

user name
******************
```

3.2 Požadavek na registraci (změnu) Text sender ID

Další třídou emailových zpráv jsou žádosti o registraci do systému, nebo změnu stávajícího sender ID. Co se obsahu týče, jsou registrační zprávy takřka identické, avšak zprávy o změně sender ID bývají velice krátké obsahující unikátní identifikátory, které nejsou z hlediska trénovaného modelu vhodné. I přes tuto skutečnost by mělo být možné tyto zprávy identifikovat, dle specifických klíčových slov.

```
From: ***** ****** <************************>
To: <sales@topefekt.com>
Date: Fri, 19 Jun 2015 03:52:55 +0000 (UTC)
Subject: Re: CART-SMS.COM - CONTACT

Hello, I bought a new credit for my customer, and details given below.
I want a senderID ASAP. Because I need to deliver this project by Monday
and still my works are pending. Hope you understand.

User: ***********************************
Need SenderID: ******

Regards,
***** ******
```

3.3 Referer ID (žádost o zařazení do affiliate programu)

Zprávy obsahující žádosti a nabídky o spolupráci s jinými vývojáři, nebo poskytovateli. Tento typ zprávy by měl být jedním z nejjednodušších ke klasifikaci, jelikož jsou zprávy krátké, stručné a obsahují specifická klíčová slova, jako "affiliate"a "referer ID".

```
From: ******* <**********************>
To: <hradilova@topefekt.com>
Date: Thu, 07 Sep 2017 11:13:54 +0100
Subject: CART-SMS.COM - CONTACT

Hello, i'm interested in becoming a part of the affiliate program.
How can we start?

*******
```

3.4 Žádost o novou proměnnou (variable)

Moduly těchto systémů používají proměnné, jenž se vkládají do textu šablon SMS zpráv. Emailová zpráva tohoto typu může obsahovat klíčová slova typu variable, nebo přímo konkrétní proměnnou. Jedná-li se však o konkrétní proměnnou může se stát, že nebude správně vyextrahovaná z textu, jelikož algoritmus pro rozdělení věty na jednotlivá slova může špatně interpretovat slova ve tvaru {variable}

```
Předmět: Variable Config
Datum: Wed, 10 Jan 2018 18:42:37 +0000
Od: ********* <******************>
Komu: support@topefekt.com

Hello topefekt team,
I need to configure a variable from a new module so when I change order
status this variable can go automatically in the sms.

Thank you, I hope you can help so can I send more sms.
```

3.5 Support request - nefunkční systém

Nyní se dostáváme ke třídě zpráv, jenž ve většině případů obsahuje velké množství textu, buďto popisující konkrétní situaci, během které se problém objevil, nebo chybový výpis aplikace. Co se chybových výpisů týče, mělo by být velice jednoduché identifikovat PHP kód, nebo výpis výjimky. Na druhou stranu anglické popisy zákazníků plné pravopisných chyb budou tvořit velice obtížně rozpoznatelný text.

```
From: ***** ******* <**********************************>
To: <sales@topefekt.com>
Date: Tue, 23 May 2017 17:48:08 +0100
Subject: CART-SMS.COM - CONTACT

Hello,
We want to manually send SMS via your simple api, what we couldn't
success on sending it. Could you please inform us what is the proper
form of the following code in your api? Could you please provide an
example? $type = "customer"; // admin x customer - senderID - from
SMS settings TAB

Thank you,
***** *******
```

3.6 Požadavek na instalaci modulu

Zprávy této třídy běžně zasílají zákazníci, jenž potřebují poradit se samotnou instalací modulu na jejich verzi systému. Obsahem běžně bývají požadavky o přesný postup instalace, nebo přihlašovací údaje na zákazníkův FTP server. Jedná se o jednu z nejčastějších emailových zpráv v tomto datasetu.

```
From: **** <**********************>
To: <sales@topefekt.com>
Date: Mon, 21 Aug 2017 20:30:34 +0100
Subject: MAGE-SMS.COM - CONTACT

Hi,
```

```
I want to subscribe for Mage SMS for my ecommerce website. please let
me know how to proceed and subscribe.

****
```

3.7 Dotaz na cenu SMS (nepochopení ceníku a kreditního systému)

Tyto emailové zprávy se obecně týkají ceny konkrétního počtu SMS zpráv, nebo zdůvodnění dané ceny. Běžně obsahují dotazy jak velké množství SMS zpráv, mohou odkoupit za uvedenou částku v dané měně. Jelikož cifry nejsou dostatečně unikátní na to, aby je bylo možné použít k rozpoznání, budou všechna čísla a částky odstraněna při tvorbě slovníku. To znamená, že zprávy této třídy budou muset být primárně rozpoznávány skrze kontext a klíčová slova jako euro, dolar a jiné.

```
From: ****** ********** <**********************>
To: <sales@topefekt.com>
Date: Mon, 17 Apr 2017 00:10:49 +0100
Subject: CART-SMS.COM - CONTACT

Respected,
I have registerd my firm on your system so I can send SMS messages.
My username is ******************. Since I am from Serbia on your site
and app it says that the price per SMS is 0.0200 euro. However when
I try to pay 5 euro of credit i see that the price per one SMS message
is 0,400 euro and that for 5 euro i can send 125 messages.
Could you clarify why the price is almost double during payment.

Best regards.
****** **********
```

3.8 Cizojazyčné zprávy

Jedna z nejtěžších tříd klasifikace jsou třídy cizojazyčné. Nejedná se však o to, že by nebylo možné natrénovat model, aby používal jiný jazyk. Klasifikační modely převádí jednotlivá slova na čísla, což znamená, že na jazyku zprávy nezáleží. Problémem je, že nikdo v této společnosti nemluví více než dvěma jazyky. Narazí-li zaměstnanci na cizojazyčnou zprávu tak požádají zákazníka o použití angličtiny. Důsledkem je malé množství zpráv v konkrétním jazyku. Pokud bychom chtěli klasifikovat zprávy do všech našich tříd pouze v ruském jazyce nebudeme mít dostatek trénovacích dat. Skutečnost je taková, že cizojazyčných zpráv je obecně velmi málo a vyskytují se hned v několika jazycích. Nejvhodnějším řešením bude rozpoznání cizojazyčné zprávy a klasifikovat ji do třídy určené k dodatečnému překladu a posléze i k potencionální klasifikaci.

```
From: ************ <***************.com>
To: sales@topefekt.com
Date: Thu, 11 Sep 2014 00:22:00 +0200
```

```
Subject: CART-SMS.COM - CONTACT

merhabalar iyi çalışmalar. müşteri siparişi verdiğinde müşteriye ve
bana sms gelsin istiyorum. 1000 sms için ödemem gereken fiyat nedir?

************
```

3.9 Požadavek na úpravu systému

Jednou z nejméně vyskytujících se zpráv v celém datasetu, jsou požadavky o modifikaci samotného modulu. Běžně se v těchto zprávách vyskytují klíčová slova, jako "customize", "modify", nebo "API".

```
From: *************** <*****************.com>
To: sales@topefekt.com
Date: Sat, 12 Dec 2015 00:07:43 +0530
Subject: RE: CART-SMS.COM - CONTACT

That is great news.

I have a small requirement. I'm using multimerch which makes my shipping
cart a market place. Can you customize the API such that one sms also
goes to the seller of the product?

Thanks
*****
```

3.10 Jiné

Poslední klasifikační třída byla určena pro zprávy, které nespadají do žádné z dříve zmíněných tříd. Nejednalo se výhradně o nežádoucí poštu, ale o zprávy obsahující obchodní nabídky, vnitrofiremní komunikaci, žádosti o pracovní nabídky, nebo přeposlané zprávy neobsahující text.

```
From: **** ***** [mailto:*********************]
To: sales@topefekt.com
Sent: Friday, April 21, 2017 11:16 AM
Subject: CART-SMS.COM - CONTACT

Hello,
I came across your plugin on the OpenCart marketplace and I'd love
to speak with you guys about your SMS usage. We have been considering
developing our own plugin for the marketplace but frankly speaking
you guys have done a great job of incorporating SMS functionality for
the the enterprises connect to OpenCart. With that in mind I'd like
to speak to you about what Silverstreet does really well which is providing
SMS connectivity.
```

```
   How do you currently source your SMS routes? Are you open to reviewing
another provider? We have been operating since 1999 out of Netherlands
and more recently (2009) we have been developing a rather strong position
in South East Asia and the surrounding regions.
Perhaps we could jump on a call to discuss things further?

Thanks and looking forward to hearing from you!

Best,
****
```

3.11 Shrnutí analýzy

Analýza zpráv ukázala, že ke klasifikaci zpráv bude potřeba deseti tříd. Byl tedy manuálně setřízen dataset obsahující 1496 zpráv (developer sada), který byl rozdělen na trénovací množinu obsahující 1192 zpráv a testovací množinu obsahující 304 zpráv. Posléze byl získán a setřízen i testovací dataset obsahující 100 zpráv (testovací sada), který byl použit k testu reálného provozu, viz kapitola 7.

Třída	Množství
Support	172
Register request sender ID	188
Referer ID	63
Variable request	34
Support request	77
Instalation request	247
SMS price	138
Foreign	47
Customization request	54
Others	476
Total	1496

Tabulka 3.1: Jednotlivé klasifikační třídy a jejich zastoupení v tréninkovém (developer) datasetu

3.12 Zpracování emailových zpráv

Hlavním účelem této aplikace je přijetí elektronické zprávy, analýza textu a její následná klasifikace do příslušné klasifikační třídy. Důležité je ale podotknout, že převážná většina analyzovaného textu bude naprosto zbytečná. Jedním z hlavních důvodů je přirozená podoba lidského jazyka. Lidská řeč obsahuje velké množství informací, které nejsou nezbytné pro identifikaci obsahu zprávy. Je tedy nutné tyto informace odstranit ještě před samotnou klasifikací.

3.12.1 Extrakce obsahu

Otevřeme-li přijatou zprávu, uvidíme vše co je k zaslání emailu potřebné. Hlavičku obsahující informace o kódování, použitou znakovou sadu, předmět zprávy, odesilatele, příjemce, verze použitých formátů, podpisy atd. Všechny tyto informace jsou důležité pro běžný provoz emailové komunikace, nicméně nás bude zajímat pouze obsah samotné zprávy a informace jako znaková sada a použité kódování.

Bohužel získaná data pocházejí z emailového serveru společnosti TOPefekt.s.r.o, což znamená, že jsou procedurálně pojmenovány a seřazeny v tom pořadí v jakém byly přijaty. Naše modely budou fungovat na principu supervised learning, což vyžaduje předběžné roztřízení našich vstupních dat dle klasifikačních tříd, které budou tvořit náš výstup. To obnáší ruční identifikaci obsahu souborů a jejich umístění do příslušného adresáře. Vstupem parseru je tedy adresářová struktura, kde každý adresář je pojmenován dle klasifikační třídy.

K extrakci emailové zprávy byla použita Python knihovna parser, díky které můžeme jednoduše přistupovat k jednotlivým informacím jako typ kódování, znaková sada, předmět, odesilatel a pro nás nejdůležitější obsah zprávy.

Je nutné podotknout, že většina emailových zpráv byla odeslána od různých emailových klientů a tedy uloženy v různých formátech. Bylo tedy nutné převést všechny zprávy do formátu txt a posléze vytvořit funkci, která vyextrahuje obsah zprávy a převedla text z různých kódování do čitelného tvaru. Nicméně i když získáme obsah v čitelném tvaru, neznamená to, že zpráva obsahuje relevantní informace.

3.12.2 Předešlá konverzace

Mnoho zpráv mimo relevantního textu obsahuje i záznam předešlé konverzace. Jelikož zprávy pochází ze stejného serveru, spousta zpráv na sebe navazuje. Otevřeme-li dvě na sebe navazující zprávy, zjistíme, že text je až na oddělovače naprosto stejný s rozdílem poslední zprávy. Jinými slovy čím více bylo na zprávu odpovězeno, tím častěji se bude obsah zpráv opakovat. Naopak pokud byla zpráva pouze přeposlána obsahuje buďto naprosto stejný text, nebo neobsahuje vůbec žádný text. V nejhorším případě se jedná o kombinaci libovolného počtu přeposlání a odpovědí, což znamená, že kupříkladu z pěti emailových zpráv získáme jen pět relevantních zpráv s velkým množstvím redundantního obsahu. Pro naše účely by bylo velice vhodné použít zprávy obsahující pouze text poslední zprávy v řadě. Vyvstává tedy otázka, zdali je vhodnější tyto předešlé záznamy konverzaci odstranit, nebo je použít pro účely klasifikace.

Ukázalo se, že by to velice zkomplikovalo klasifikaci, jelikož spousta zákazníků této firmy nezasílá nový email v případě nového dotazu, což znamená, že téma zprávy (klasifikační třídy) se mění od zprávy ke zprávě. Abychom tedy mohli jednoznačně určit, do jaké třídy patří konkrétní email, musíme klasifikovat pouze poslední zaslanou zprávu, viz obrázek 7.1.

3.12.3 Stopslova

Dle Zipfova zákona [8] v každém přirozeném jazyce (tj. jazyce vzniklém přirozeným vývojem) existuje jisté rozdělení četnosti výskytu určitých slov. Slova s nejvyšším výskytem, které mají rank 1, až po slova s rankem n, která označují slova s nejmenším výskytem v daném textu. Tyto četnosti mají tvar exponenciální funkce. Slova na počátku této křivky jsou převážně spojky, zájmena, neurčité členy apod. Jinými slovy se jedná o slova, která se vyskytují velice často a nesdělují žádné kritické informace. Je nutné podotknout, že odstra-

není stopslov nezaručí maximální relevanci textu. Tato relevance silně závisí na kontextu zprávy, osobitému stylu autora, nebo použití specifických termínů.

I have many customers registering on my Opencart Website, but abandonment rate is extremely high, most probably due to shipping costs.
↓
I many customers registering Opencart Website, abandonment rate extremely high, probably due shipping costs.

3.12.4 Stemmatizace

Jakmile jsou odstraněna všechna přebytečná slova je potřeba identifikovat zbytek, což se může jevit problematické vlivem různých tvarů slov. Stemmatizace je nalezení kmene slova, běžně používané u vyhledavačů, které dovolují vyhledávat bez ohledu na konkrétní tvar. Abychom tedy nemuseli pracovat s duplicitními záznamy slov, které se liší pouze v jednom symbolu vlivem skloňování, nebo množného čísla aplikujeme na všechna získaná slova stemmatizaci [1]. Pro provedení stemmatizace byl použit PorterStemmer z knihovny nltk.

I'd like to install this extension and need to know how to activate credits
↓
i'd like to instal thi extens and need to know how to activ credit

3.12.5 Nepoužitelná slova

I přes stemmatizaci, odstraněná stopslova a výběru pouze aktuální zprávy, náš text stále obsahuje informace, které jsou nám naprosto k ničemu. Mnoho uživatelů používá personalizované hlavičky a patičky obsahující propagaci, slogany, loga, dodatečné kontaktní informace, nebo vlastní css styly. Vše zmíněné je obsah, který by nám umožnil identifikovat konkrétního uživatele, ale při identifikaci třídy nijak nepřispívá. V nejhorším případě se může jednat o css styly. Css styly velice často obsahují klíčová slova a číselné konstanty, které činí téměř 80% obsahu zprávy. Nicméně existují způsoby jak se tohoto obsahu zbavit.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"
"http://www.w3.org/TR/=REC-html40/strict.dtd»
<html><head><meta name=3D"qrichtext"content=3D"1"/>
...
```

Jedním z možných způsobů je použít slovník povolených slov. Pokud narazíme na výraz, který nepatří do slovníku, můžeme jej přeskočit. Tento způsob ale vyžaduje použití objemných slovníků a náš dataset obsahuje i slova cizojazyčná což by obnášelo použití hned několika slovníků. Toto řešení ovšem nepočítá s textem, který není správně gramaticky zapsaný. Pokud bychom takto redukovali zprávy od nerodilého mluvčího, eliminovali bychom hned polovinu obsahu. Zdali by to bylo prospěšné či škodlivé je nejisté. Na jednu stranu odstraňujeme nepoužitelná slova, ale na druhou stranu se jedná o běžnou podobu zpráv.

Nicméně i když se nám nepodaří odstranit všechna nepotřebná slova, stále můžeme minimalizovat jejich vliv na trénink modelu.

3.12.6 Slovník

Nyní když jsme izolovali relevantní informace z daného textu, můžeme tyto informace použít k vytvoření slovníku, vypočtením četnosti výskytů jednotlivých termínů. Na pořadí termínu ve své podstatě ani nezáleží, slovník slouží pouze jako vyhledávací tabulka unikátních termínů.

Nevýhodou slovníku vytvořeného za pomoci spektrální analýzy je, že termín se stane významným pouze tím, že je zmíněn v dostatečně velkém množství. Pokud bychom natrénovali náš klasifikátor na takto vytvořeném slovníku, vystavujeme se nebezpečí související se špatně zvolenou trénovací sadou. Trénovací sada by mohla kupříkladu obsahovat jednu zprávu s bezvýznamným avšak často se vyskytujícím termínem, který se v jiné zprávě už nikdy neobjeví. Model natrénovaný touto datovou sadou by byl posléze tímto bezvýznamným termínem ovlivňován. Tento problém může být řešen vhodně zvolenou trénovací sadou, nebo metodou popsanou později.

Funkce pro tvorbu slovníku byla navržena tak, aby dynamicky procházela adresářovou strukturu, za účelem umožnění potenciálního rozšíření trénovacích dat. Obsah každé emailové zprávy je rozdělen na jednotlivé věty a následně na jednotlivá slova. Pro správnou reprezentaci slovníku musela být odstraněna stop slova, speciální znaky a čísla. Dalším ne nutně nezbytným krokem bylo odstranění speciálních jmen, kterými jsou jména zaměstnanců firmy a jejich emailové adresy. Je zřejmé, že slova tohoto typu se budou velice četně objevovat a klasifikátoru neposkytnou žádné dodatečné informace ohledně kontextu.

Na závěr jsou slova převedena do kořenového tvaru a četnost jejich výskytu je uložena do struktury slovníku definované jako kolekce uspořádaných dvojic. Výsledný slovník obsahuje seznam nejčastěji vyskytujících se slov, který je posléze lexikograficky seřazen.

Kapitola 4

Klasifikace

Jelikož máme k dispozici přes šest tisíc emailových zpráv, můžeme vytvořit náš klasifikační model. K tomuto účelu využijeme několik běžně užívaných metod, což zahrnuje i strojové učení. Trénink našeho klasifikátoru bude využívat principu supervised learning, což je učení s učitelem. Je tomu tak, protože model během tréninku ví, jaká data používáme k učení a co se učí. Naším cílem je na základě obsahu zařadit zprávu do jedné z K tříd, alternativně do N z K tříd. Před samotnou klasifikací provedeme redukci dimenzionality, s cílem lepšího zobrazení vstupních dat.

4.1 Redukce dimenzionality

Redukce dimenze představuje důležitý krok statistické analýzy. Jedná se o extrakci informací z mnohorozměrných dat, která může být v případě vysoce dimenzionálních dat zcela nezbytná.

Existuje řada metod, které zjednodušují proces různých typů analýz. Může se jednat o klasifikační, shlukové analýzy a jiné. Tyto metody redukující dimenzionalitu umožňují přímou extrakci důležitých informací z dat, popisují rozdíly mezi skupinami dat, zdůrazňují příspěvek jednotlivých proměnných v daných skupinách dat. Z anglicky psaných knih můžeme k danému tématu doporučit [10], anebo [5], [6], kde jsou tytéž metody diskutovány spíše z hlediska dolování znalostí (data mining). Z česky psaných zdrojů můžeme doporučit knihu [7].

Existují tří základní principy jak předzpracovat data pro účely strojového učení.

- **Feature extraction** - Transformace původních dat do takového tvaru, který je vhodný pro daný model.
- **Feature transfomation** - Transformace pro zvýšení přesnosti použitého algoritmu.
- **Feature selection** - Selekce pouze relevantních dat, nebo jinými slovy odstranění zbytečných dat.

Podle jiného kritéria dělíme metody pro redukci dimenze na supervidované a nesupervidované. Supervidovanými jsou takové, které jsou určeny pro data pocházející ze dvou, nebo více skupin a současně využívající informaci o tom, které pozorování patří do které skupiny. To umožňuje zachovat oddělitelnost mezi skupinami. Někteří autoři varují, že analýza hlavních komponent jako příklad nesupervidovaných metod není vhodná pro redukci

dimenze dat pocházejících ze dvou nebo více skupin v situaci, kdy cílem je klasifikační analýza. V našem případě bude selekce prováděna během vytváření slovníku a extrakce během tvorby feature vektoru.

4.2 Term Frequency-Inverse document frequency

Term Frequency-Inverse document frequency (dále jen Tf-idf) je metoda určující, jak je termín daného dokumentu důležitý v kontextu celého datasetu. Jinými slovy čím více se slovo vyskytuje v celé sadě dokumentů, tím méně důležitější je. Pokud se ale všechny výskyty konkrétního termínu nacházejí v rámci jednoho dokumentu, relevance daného slova stoupne. Výsledná hodnota významnosti termínu je spočtena jako součin následujících dvou rovnic 4.1, 4.2.

4.2.1 Term Frequency (Tf)

Frekvence termínu udává přesně, co název vypovídá. Jedná se o pouhý výčet výskytů konkrétních slov v rámci jednoho dokumentu. Je však nezbytné tuto hodnotu normalizovat, jelikož v dlouhých dokumentech se hledaný výraz může vyskytovat častěji než v kratších. Délka dokumentu tedy nepřímo ovlivňuje relevantnost dokumentu.

$$tf_{i,j} = \frac{n_{i,j}}{\sum^{k} n_{k,j}} \tag{4.1}$$

4.2.2 Inverse document frequency (Idf)

Inverse document frequency určuje váhu běžnosti výskytu termínu skrze celou sadu dokumentů. Jinými slovy čím častěji se termín vyskytuje v trénovací sadě dokumentů, tím méně důležitý termín je. Naopak čím méně se termín vyskytuje, tím důležitější bude. Pokud bychom neodstranili již zmíněná stopslova, bylo by jim touto metodou přiřazeno velmi malé skóre.

$$idf_i = log \frac{\mid D \mid}{\mid \{j > t_i \in d_j\} \mid} \tag{4.2}$$

4.3 Statistická klasifikace

Statistická klasifikace funguje na principu vyhledávání vzorů. Samozřejmě mluvíme o velice širokém pojmu, ale v našem případě se jedná o zařazení zprávy do klasifikační třídy na základě podobnosti terminologie jiných zpráv. Jinými slovy obsahuje-li zpráva podobná slova, je velice pravděpodobné, že budou patřit do stejné klasifikační třídy.

4.3.1 Naïve Bayesian model

Ke klasifikaci existuje mnoho metod, jednou z nich je Naïve Bayesian model. Jedná se o jeden z nejpoužívanějších modelů určených ke klasifikaci dat [3]. Přestože se jedná o jeden z nejjednodušších klasifikačních modelů, je stále velmi efektivní. Nejběžnějším typem klasifikace je binární klasifikace neboli klasifikace do dvou tříd. Uvedeme si jednoduchý příklad klasifikace emailových zpráv na vyžádanou a nevyžádanou poštu. Pravděpodobnost, že se jedná o spam zprávu je vypočtena jako pravděpodobnost nalezení komponent feature

vektoru ve spamové zprávě $p(C_k)$. Tato pravděpodobnost je vynásobena pravděpodobností výskytu spamu v sadě testovaných zpráv $p(x|C_k)$. Tento součin je následně vydělen obecnou pravděpodobností výskytu $p(x)$ konkrétní komponenty feature vektoru v textu 4.3.

$$p(C_k|x) = \frac{p(C_k)p(x|C_k)}{p(x)} \tag{4.3}$$

Během tréninku jsou feature komponenty každého dokumentu zaznamenány. Pokud se jedná o spam, pak je feature přidán, jak do pravděpodobnosti spamu, tak do obecné pravděpodobnosti. Featury běžných zpráv jsou přidány pouze do obecné pravděpodobnosti.

Pravděpodobnost, že bude jakýkoli text označen za spam, je určen parametrem algoritmu. Čím větší pravděpodobnost tím vyšší bude počet zpráv klasifikovaných jako spam. Zvýšení této hodnoty sníží počet falešných negativ (false negative - spam označený jako legitimní zpráva), ale taktéž zvýší počet falešných pozitiv (false positive - legitimní zpráva označená jako spam).

4.4 Funkcionální klasifikace

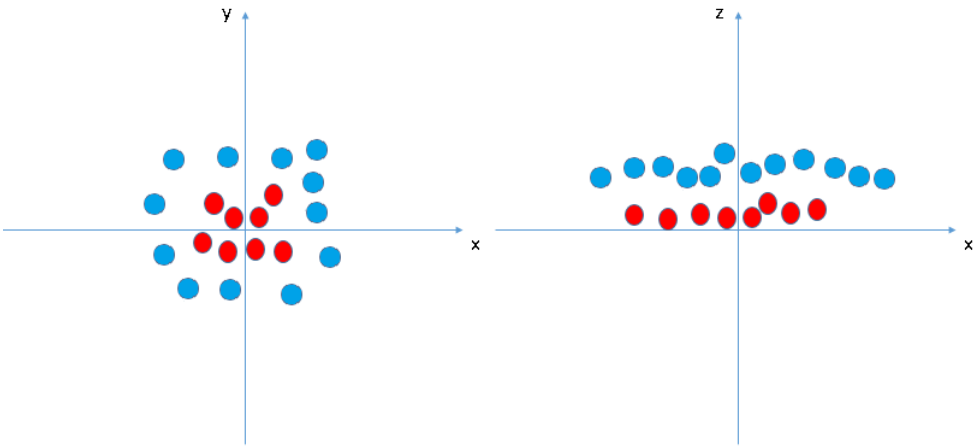
Pokud nahlížíme na každé číslo ve feature vektoru jako na koordinát dimenze, pak se každý dokument dá reprezentovat jako bod v multidimenzionálním prostoru, kde počet dimenzí je roven počtu hodnot ve feature vektoru. Tato reprezentace nám umožní použít geometrické metody klasifikace.

4.4.1 K-nejbližších sousedů

Jednou z nejjednodušších geometrických metod je k-nejbližších sousedů (kNN klasifikátor) [9]. Funguje na velice jednoduchém principu. Klasifikovaný dokument je reprezentován jako bod ve vícedimenzionálním prostoru a hledá pro něj k-nejbližších sousedů. Pokud všechny patří do stejné klasifikační třídy, bude tedy náš dokument taktéž patřit do této třídy. Pokud tedy 4 z 5 nejbližších sousedů patří do třídy A poslední z nich patří do třídy B, klasifikujeme zkoumaný dokument do třídy A s jistotou 80%. Tato metoda je také velice modifikovatelná a poskytuje široký rozsah vylepšení přesnosti.

4.4.2 Support vector machines SVM

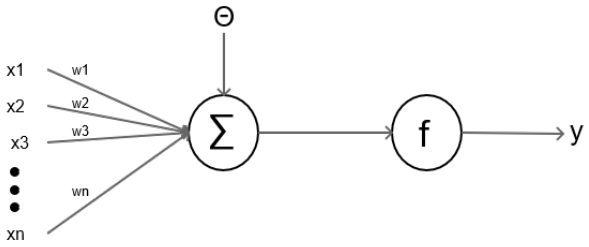
Současně jedním z nejvíce zkoumaných klasifikátorů je SVM. Opět vizualizujeme na příkladu dvou tříd. Představíme-li si třídy jako dva shluky bodů a představíme si hranice mezi těmito třídami jako bublinu, můžeme poté identifikovat dokumenty, které reprezentují hraniční body každé třídy. Najdeme-li vektor, který prochází skrze tyto hraniční body v prostoru, tak aby všechny tyto dokumenty stejné kategorie byly na stejné straně tohoto vektoru, nalezli jsme vektor nazývaný podpůrný vektor. Z matematického hlediska, zprůměrujeme-li dva podpůrné vektory ze dvou tříd, můžeme určit vektor, který bude ležet zhruba mezi těmito kategoriemi. Následně tedy určíme, o jaký dokument se jedná v závislosti na jeho pozici vůči danému vektoru. Metoda se umí vypořádat i s případy, kdy data v prostoru nejsou jednoduše separovatelná. Místo lineární funkce můžeme použít hyperplochu v tří dimenzionálním prostoru a separovat zdánlivě neseparovatelné třídy, viz obrázek 4.1. SVM k tomuto účelu používá kernely, což jsou funkce, které převádí neseparovatelné třídy na separovatelné zvýšením dimenze prostoru. Testy s různými kernely ukázaly, že naše data jsou dostatečně separabilní pro použití lineárního kernelu.



Obrázek 4.1: SVM - Lineárně neseparovatelná data (vlevo) Separovatelné ve tří dimenzionálním prostoru (vpravo)

4.5 Klasifikace za pomoci neuronové sítě

Neuronová síť je masivní paralelně distribuovaný procesor složený z jednotlivých výpočetních jednotek zvaných neurony, viz 4.2, které umožňují schraňovat informace z předchozích událostí pro opětovné použití. Síť nabývá znalostí, jak skrze učení na základě vstupních hodnot, tak učení na základě vnitřních skrytých stavů.



Obrázek 4.2: Činnost neuronu

Principiálně je užití neuronových sítí ke klasifikaci poměrně jednoduchý proces. Neuronové síti je poskytnut feature vektor na vstupu a kategorizace se objeví na výstupu. Každý z výstupů má přidělenou klasifikační třídu. Výše jednotlivých výstupních hodnot znázorňuje s jakou pravděpodobností je si síť jistá svým rozhodnutím. Pokud výstupní signál nedosáhne cílové hodnoty, může být klasifikace odmítnuta, aby bylo docíleno vyšší spolehlivosti výsledku.

Výkonným prvkem se zde rozumí jeden neuron. Ten zpracovává hodnoty na svém vstupu podle následujícího vztahu 4.4.

$$[H]y = f(\sum_{i=1}^{N} \omega_i x_i + \Theta) \tag{4.4}$$

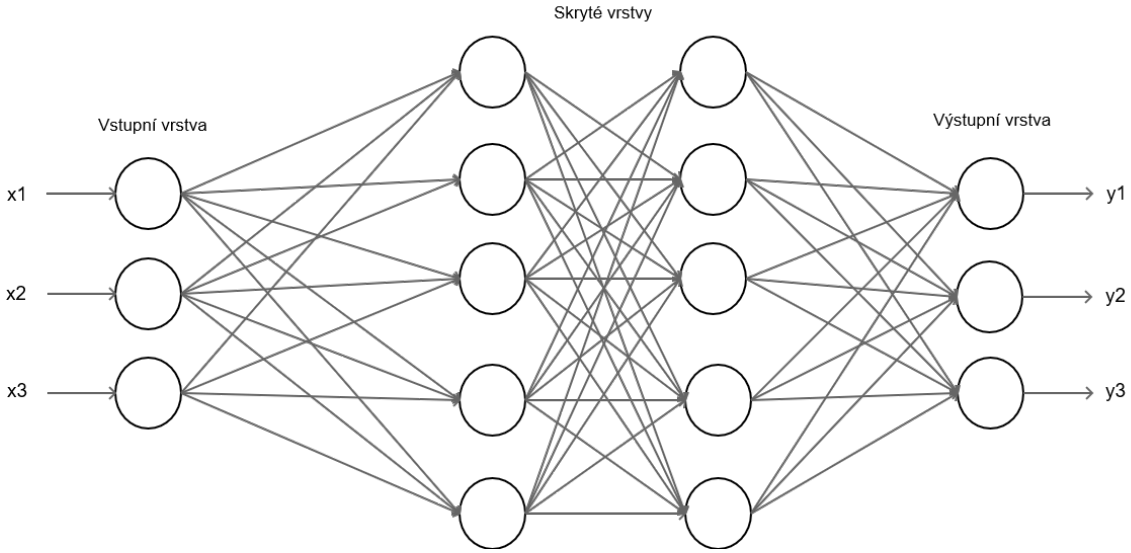
V rovnici 4.4 kde f je přenosová funkce neuronu, x_i jsou vstupy neuronu, ωi i jsou váhy vstupů a Θ je práh neuronu. Suma vážených vstupů s přičteným prahem se též označuje

jako vnitřní potenciál neuronu. Aktivační (přenosová) funkce neuronu převádí vnitřní potenciál neuronu do definovaného oboru výstupních hodnot. Nejčastější aktivační funkce jsou lineární funkce, sigmoid, skoková funkce, hyperbolický tangens a mnoho jiných.

Nicméně hlavní problém při použití neuronových sítí je samotný návrh sítě. Teoreticky je možné zkonstruovat neuronovou síť jakkoli komplexní, ale je velmi obtížné předpovědět, zdali bude daná síť vynikat v daném klasifikačním problému. Ukázalo se, že prozatím je nejpraktičtější použít neuronové sítě na klasifikaci textu, což je také náš případ.

Při učení s učitelem se neuronová síť učí srovnáváním aktuálního výstupu s výstupem požadovaným a nastavováním vah tak, aby se dosáhlo co nejmenšího rozdílu mezi skutečným a požadovaným výstupem. Různých typů umělých neuronových sítí je mnoho (například Perceptron, Hopfieldova síť, Kohonenova síť, ART síť atd.). V této práci však budou uvažovány pouze vícevrstvé neuronové sítě, neboli perceptron [2] [4] [9].

Jak název vypovídá, vícevrstvé sítě se skládají z vrstev neuronů, vrstvy jsou mezi sebou vzájemně propojeny tak, že neuron z jedné vrstvy je propojen se všemi neurony následující vrstvy, přitom v rámci vrstvy nejsou definována žádná propojení 4.3.



Obrázek 4.3: Vícevrstvá neuronová síť se dvěma skrytými vrstvami

Počet neuronů vstupní vrstvy odpovídá velikosti vstupu a analogicky počet neuronů výstupní vrstvy je postaven na kódování výstupu. V našem případě bude počet výstupních neuronů odpovídat počtu klasifikačních tříd. Nadále mezi vstupní a výstupní vrstvou se obecně nachází několik skrytých vrstev neuronů. Aby byla naše síť schopná naučit se daný problém, je nezbytné zvolit vhodný počet neuronů ve skrytých vrstvách. Při nízkém počtu neuronů ve skrytých vrstvách má za následek neschopnost sítě se naučit daný problém, neboť síť nemá dostatečnou paměťovou kapacitu. Naopak použijeme-li příliš velký počet neuronů, může dojít k přeučení sítě.

Učení neuronové sítě lze formálněji popsat následujícím způsobem. Uvažujme neuronovou síť s n vstupy a m výstupy. Po této síti požadujeme realizaci zobrazení Φ z množiny vstupních vektorů $x \subset R^n$ do množiny výstupních vektorů $Y \subset R^m$. Aproximaci tohoto zobrazení provede neuronová síť za pomoci funkce.

$$\overrightarrow{y} = f(\overrightarrow{x}, \overrightarrow{\omega}\overrightarrow{\Theta}) \tag{4.5}$$

V rovnici 4.5 $\overrightarrow{y}$ je výstupní vektor, $\overrightarrow{x}$ je vstupní vektor, $\overrightarrow{\omega}$ je vektor všech vah sítě, $\overrightarrow{\Theta}$ je vektor všech prahů a f je aktivační funkce. Učící algoritmus nalezne pro požadované zobrazení $\Phi : X \to Y$ takové parametry $\overrightarrow{\omega}$ a $\overrightarrow{\Theta}$ funkce f, že tato funkce je právě aproximací tohoto zobrazení.

Nejčastějším algoritmem používaným v oblasti neuronových síti pro klasifikaci a učení neuronové sítě je algoritmus zpětného šíření chyby (Back-propagation). Nejdříve v rámci algoritmu inicializujeme váhy jednotlivých neuronů na náhodné hodnoty, načež je vypočtena odezva sítě pro náhodný vstup z tréninkové sady. Následně se výsledek porovná s požadovaným výstupem. Rozdílem těchto hodnot nám umožní určit parciální chybu sítě. Celková chyba sítě je následně dána součtem parciálních chyb všech neuronů. Takto získaná parciální chyba je posléze vynásobena tzv. rychlostí učení (learning-rate) a následně propagována zpět skrze síť od výstupů ke vstupům. Během šíření vypočtené chyby jsou upravovány váhové koeficienty, což má za následek snížení chyby v následující iteraci. Algoritmus tedy funguje na principu v nalezení minimální hodnoty chybové funkce při modifikaci vah neuronů. Tato funkce může být vizualizována jak zakřivenou plochu v hyperprostoru. Hodnota vah a prahů je následně pak zobrazena jako bodem na této ploše.

4.5.1 Implementace

Naše neuronová síť byla implementovaná pomocí knihovny pro strojové učení scikit-learn. Síť bude obsahovat n vstupních neuronů, kde n je délka vstupního vektoru. Tvorba vstupního vektoru je popsána v následující kapitole 5.2. Aktivační funkce vstupní vrstvy je funkce relu. Během vývoje bylo experimentováno s různým počtem skrytých vrstev o různých velikostech. Nakonec byla použita jedna skrytá vrstva o velikosti 512 neuronů, kde aktivační funkce každého neuronu byla funkce sigmoid. Výstupní vrstva musela mít počet neuronů roven počtu klasifikačních tříd. Za aktivační funkci výstupní vrstvy byla zvolena funkce softmax.

Kapitola 5

Trénink a metriky

Všechny modely zmíněné v předchozí kapitole musí být natrénovány a posléze objektivně otestovány. V této kapitole si tedy popíšeme některé způsoby zobrazení výsledků klasifikace společně s použitým formátem vstupních dat a metrik, které použijeme k hodnocení výstupů jednotlivých modelů.

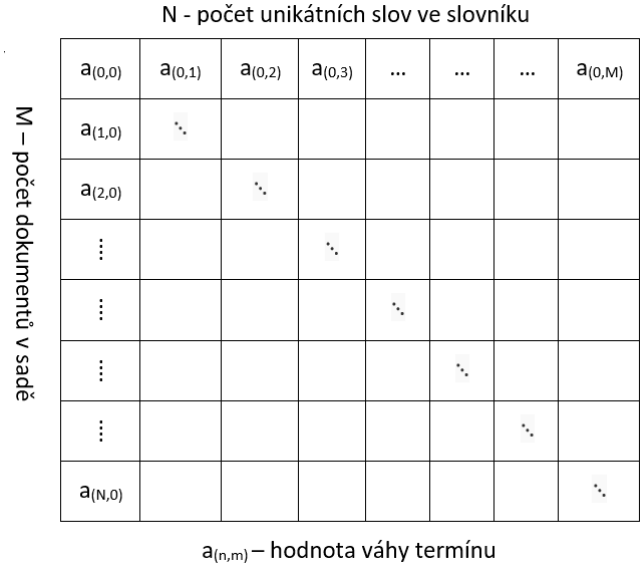
5.1 Klasifikační modely

Aplikace byla vytvořena tak, aby bylo možné přepínat mezi jednotlivými modely, kterými jsou Naïve Bayes, SVM, K-Neighbors a námi specifikovaná neuronová síť. Ještě před samotným tréninkem a predikcí bylo testováno, zdali v modelu nedochází k přetrénování. Byla tedy použita metoda cross_val_score(), jenž iterativně rozdělí poskytnutá data na tréninkovou sadu a validační sadu. A posléze s každou iterací natrénuje a ohodnotí model za pomoci F1 skóre.

Výstupem každé klasifikační metody je pole obsahující index nejpravděpodobnější třídy, který je použit v poli obsahující všechna jména klasifikačních tříd. Je tomu tak u každé metody kromě naši neuronové sítě, jejichž výstupem je pole indexované jednotlivými emailovými zprávami každé, z nichž obsahuje další pole, uvnitř kterého jsou zaznamenány pravděpodobnosti pro jednotlivé klasifikační třídy. Jako výslednou klasifikační třídu zvolíme tu s nejvyšší pravděpodobností.

5.2 Tvorba tréninkové matice

Abychom byli schopni natrénovat naše modely, musíme jim poskytnout tréninkovou matici. Matice obsahuje hodnoty určující vztah mezi unikátními slovy a jednotlivými dokumenty. Tuto matici si můžeme představit jako matici s dimenzemi NxM, kde dimenze N je definovaná počtem unikátních slov ve slovníku. Zatímco M reprezentuje počet všech dokumentů, jenž byly použity k tvorbě tohoto slovníku. Znamená to tedy, že každý dokument je reprezentován jako N-dimenzionální vektor, kde každá proměnná vektoru reprezentuje unikátní slovo ve slovníku. Jednotlivé hodnoty matice reprezentují váhu daného termínu v dokumentu, viz obrázek 5.1.



Obrázek 5.1: Tréninková matice - Řádek matice určuje všechny unikátní slova ze slovníku. Sloupce určují všechny dokumenty obsažené v tréninkové sadě. Konkrétní buňky obsahují buďto hodnotu četnosti výskytu daného slova v celém korpusu, nebo hodnotu Tf-idf daného slova, vůči celému korpusu.

5.3 Hodnocení modelu

Dále nám vyvstává otázka. Jak velké množství dat musíme modelu poskytnout, než bude schopen spolehlivě klasifikovat? Obecně platí pravidlo, čím více informací modelu poskytneme, tím lépe bude fungovat. Jak již bylo řečeno, existuje takzvané přeučení sítě nevhodnou volbou počtu a nastavení neuronů ve skrytých vrstvách. Tak či tak je nezbytné naše klasifikační metody nějakým vhodným způsobem objektivně ohodnotit. K tomuto účelu se používají dvě metriky nazývané:

- *Precision* - počet správných výsledků děleno počtem všech vrácených výsledků.
- *Recall* - počet správných výsledků děleno počtem všech výsledků, které měly být vráceny.

Než ale začneme počítat tyto metriky, musíme si připravit takzvanou confusion matici. Abychom mohli ohodnotit úspěšnost metody klasifikující do desíti tříd, vytvoříme matici 10x10, ve které jsou klasifikovaná data zobrazena následovně. Každý řádek matice určuje dokumenty, které patří do dané klasifikační třídy. Každý sloupec matice určuje dokumenty, které model klasifikoval do dané třídy. Z toho vyplývá, že správně klasifikované soubory jsou ty, jenž leží na hlavní diagonále matice. Čteme-li tedy řádek matice, všechny dokumenty mimo hlavní diagonálu jsou dokumenty, které měly být klasifikovány do třídy daného řádku, ale byly umístěny do třídy dané sloupcem hodnoty viz obrázek 5.2.

Predikované třídy

Skutečné třídy	A	B	C	D	E	F	G	H	I
A	TA	FB	FC	FD	FE	FF	FG	FH	FI
B	FA	TB	FC	FD	FE	FF	FG	FH	FI
C	FA	FB	TC	FD	FE	FF	FG	FH	FI
D	FA	FB	FC	TD	FE	FF	FG	FH	FI
E	FA	FB	FC	FD	TE	FF	FG	FH	FI
F	FA	FB	FC	FD	FE	TF	FG	FH	FI
G	FA	FB	FC	FD	FE	FF	TG	FH	FI
H	FA	FB	FC	FD	FE	FF	FG	TH	FI
I	FA	FB	FC	FD	FE	FF	FG	FH	TI

<T/F><Class> (TB = True B, FB = False B)

Obrázek 5.2: **Confusion matice - True A**-Zpráva A zařazena správně do třídy A. **False A**-Zpráva B zařazena do třídy A.

5.4 Metriky a měření

Nyní se dostáváme k měření, pro které využijeme výše zmíněné metriky precision a recall. Jedná se o velice jednoduché metriky, které se často používají jako součást komplexnější metriky jako je F1 skóre.

Nejdříve ale musíme provést měření, které v nejjednodušších případech strojového učení s učitelem (supervised learning) probíhá následovně. Všechna data se náhodně rozdělí do tří sad. Trénovací sada, testovací sada a nakonec validační sada. Trénovací data jsou náhodně vložena na vstup neuronové sítě a veškeré chyby jsou opraveny za pomocí zpětné propagace (backpropagation). Data jsou vložena do sítě hned několikrát, dokud hodnota recall poslední iterace není vyšší než hodnota námi zvoleného prahu, nebo pokud nedosáhne námi zvoleného počtu iterací.

$$Precision = \frac{TP}{TP + FP} \tag{5.1}$$

Ve chvíli kdy je náš model natrénován, následuje testování. Všechna testovací data jsou přivedena na vstup klasifikátoru a hodnota precision 5.1 je vypočtena. Pokud je hodnota precision nižší než předem zvolená hodnota (minimum generalisation threshold - většinou kolem 70-80%) vrátíme se zpět k fázi trénování (ve většině případů na pevný počet iterací).

$$Recall = \frac{TP}{TP + FN} \tag{5.2}$$

Následuje verifikační fáze, která je velice podobná testovací. Spočívá ve výpočtu průměrné odchylky skrze celou datovou sadu. Co je na této fázi jedinečné, je její výsledek. Jedná se o F1 skóre 5.3 a jedná se o výslednou hodnotu, kterou lze porovnávat jednotlivé metody. Pokud je výsledek metriky dostačující, je síť hotová. Můžeme tedy ukončit učení a začít ji používat k cílenému účelu. Nicméně pokud hodnota F1 skóre dostačující není, nehledě na to jak dlouho budeme síť učit, výsledek nezlepšíme. Pokud bychom v této fázi chtěli vylepšit výsledek, je nutné změnit celý model.

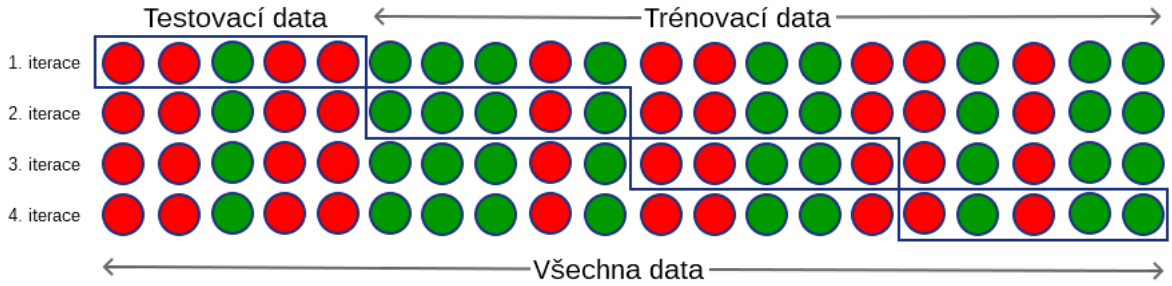
$$F1score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{5.3}$$

Další metrikou, která nám poskytuje celkem dobrý přehled o tom, zdali model klasifikuje s dostatečnou přesnosti je Accuracy, která jednoduše udává, kolik vzorků bylo správně klasifikováno.

5.5 K-fold cross-validation

Často využívanou metodou pro verifikaci modelu je metoda k-fold cross validation, která existuje v několika různých variantách. Základní varianta funguje tak, že data jsou náhodně rozdělena do k disjunktních množin přibližně stejné velikosti viz obrázek 5.3. Vyhodnocení probíhá v k iteracích, při kterých je použita jedna část pro trénink a zbytek pro testování. Výsledky jednotlivých iterací jsou následně zprůměrovány pro dosažení vyšší přesnosti.

Následuje metoda leave-one-out, která funguje na principu, použij všechny vzorky na trénink, ale ponech jeden na testování. Přes tento proces iterujeme tak dlouho, dokud máme vzorky. Jedná se o efektivní metodu, ale je tomu tak pouze pokud nemusíme zpracovat přehnaně velké množství vzorků. Existuje ještě metoda zvaná stratified cross-validation, která rozdělí data do jednotlivých podmnožin s přibližně stejnou distribucí tříd.



Obrázek 5.3: K-fold pro k=4

Zbývá nám metoda bootstrap, která je velice vhodná pro práci s velkým množstvím vzorků, neboť vybírá jednotlivé vzorky náhodně. Jelikož se jedná o náhodný výběr, může na rozdíl od již zmíněných metod vybrat konkrétní vzorek více než jednou. Pracujeme-li tedy s množinou velikosti n, provedeme n-krát náhodný výběr n trénovacích vzorků. Přitom pravděpodobnost výběru vzorku je $\frac{1}{n}$, z čehož vychází, že pravděpodobnost nevybrání vzorku bude rovna $1 - \frac{1}{n}$. Pokud tedy provedeme n výpočtů, při kterém vzorek vybrán nebude, bude výraz vypadat takto 5.4.

$$\lim_{n\to\infty}(1 - \frac{1}{n})^n = e^{-1} = 0.367 \tag{5.4}$$

Znamená to tedy, že v rámci této metody bude zvoleno kolem 63% tréninkových dat a 37% testovacích dat.

Tak či tak bude tedy vhodné experimentovat s různými metodami rozdělení testovacích a trenovacích dat pro dosažení optimálních výsledků.

Kapitola 6

Experimentování a testování

Míra přesnosti klasifikačního modelu vyjadřuje jeho schopnost klasifikovat neznámá data. Určení přesnosti je nutné provádět na neznámých datech (takových jenž nebyly použity pro trénování), opačný případ by vedl ke správným výsledkům, ale chybnému modelu. Vyhodnocení je pak založeno na testování naučených znalostí na datech, pro které máme možnost porovnat, zda se nalezené výsledky shodují s realitou.

Jelikož nevíme, jak se budou dané modely chovat za použití konkrétního datasetu, proto musíme provést měření modelů bez jakýchkoli modifikací pouze za použití námi vytvořeného slovníku popsaného v předešlých kapitolách.

6.1 Naïve Bayes

Za účelem otestování základní přesnosti modelu musíme zjistit jaký vliv mají různé typy trénovacích vektorů na náš model. Pro začátek byly provedeny testy s délkou vektoru, viz tvorba tréninkové matice 5.2. Bylo experimentováno s délkami vektoru v rozsahu 100 - 6000. Dále byly provedeny testy s hodnotami vektoru, jenž zobrazovaly frekvenci výskytu slova, nebo hodnotu Tf-idf.

Od začátku je zřejmé, že délka vektoru bude ovlivňovat přesnost, což potvrzují i výsledky v tabulce 6.1. Další testy ukázaly, že přesnost modelu se dále nezlepšuje od délky vektoru 500. Co se týče metody použité k tvorbě vektoru (Frekvence, Tf-idf), zaznamenali jsme drobné zlepšení v některých klasifikačních třídách, nicméně se jedná o velmi malé nárůsty a při globálním pohledu na všechny třídy se spíše jedná o náhodný rozptyl.

6.2 SVM

V rámci konzistence jsme podstoupili model SVM stejnému testu, abychom stanovili základní přesnost, vůči které budeme posléze porovnávat výsledky dalších testů zobrazené v tabulce 6.2. Stejně jako u modelu Naïve Bayes délka vektoru ovlivňuje přesnost modelu, která se od určité délky ustálí a dále neroste. Dosavadní měření ukazuje, že oproti metodě Naïve Bayes, metoda SVM si počíná o něco lépe, co se přesnosti týče i za použití nevhodné délky vektoru.

Porovnáme-li metody z hlediska hodnot vektoru, zdá se, že metoda Tf-idf opět nepřinesla žádné razantní zlepšení.

Dev. test set	Freq (100)	Tf-idf (100)	Freq (6000)	Tf-idf (6000)	Množ.
Support	0.46	0.53	0.51	0.51	35
Registration_req.	0.63	0.59	0.67	0.63	38
Referer_id	0.62	0.72	0.78	0.78	13
Variable_req.	0.47	0.55	0.62	0.50	7
Support_req.	0.56	0.64	0.55	0.42	16
Module_instalation	0.62	0.70	0.62	0.61	50
Price_query	0.69	0.72	0.81	0.72	28
Foreign	0.00	0.00	0.82	1.00	10
Others	0.77	0.76	0.77	0.80	96
Customization_req.	0.36	0.00	0.00	0.00	11
Average	0.62	0.63	0.66	0.66	
Accuracy	190	198	**203**	202	304

Tabulka 6.1: **Naïve Bayes - Base**. Každý řádek popisuje jednotlivé klasifikační třídy Každý sloupec definuje přesnost metody. **Freq**-použití frekvenčního slovníku během tréninku metody. **Tf-idf**-použití Tf-idf metriky během tréninku metody. **Vektor** délka tréninkového vektoru (100)/(6000). **Množ.**-množství testovacích dat v dané klasifikační třídě. **Average**-Průměrná přesnost. **Accuracy**-počet správně klasifikovaných zpráv.

Dev. test set	Freq (100)	Tf-idf (100)	Freq (6000)	Tf-idf (6000)	Množ.
Support	0.50	0.53	0.60	0.61	35
Registration_req.	0.63	0.66	0.65	0.61	38
Referer_id	0.48	0.54	0.77	0.75	13
Variable_req.	0.57	0.62	0.36	0.40	7
Support_req.	0.57	0.62	0.43	0.50	16
Module_instalation	0.64	0.64	0.66	0.64	50
Price_query	0.73	0.75	0.79	0.73	28
Foreign	0.18	0.18	0.59	0.59	10
Others	0.80	0.79	0.82	0.81	96
Customization_req.	0.11	0.20	0.38	0.32	11
Average	0.64	0.65	0.69	0.68	
Accuracy	198	202	**211**	209	304

Tabulka 6.2: **SVM - Base**. Každý řádek popisuje jednotlivé klasifikační třídy Každý sloupec definuje přesnost metody. **Freq**-použití frekvenčního slovníku během tréninku metody. **Tf-idf**-použití Tf-idf metriky během tréninku metody. **Vektor** délka tréninkového vektoru (100)/(6000). **Množ.**-množství testovacích dat v dané klasifikační třídě. **Average**-Průměrná přesnost. **Accuracy**-počet správně klasifikovaných zpráv.

6.3 K-nejbližších sousedů

Během testování se ukázalo, že metoda K-nejbližších sousedů, není nejvhodnější pro daný dataset. Podíváme-li se na naměřené výsledky v tabulce 6.3 uvidíme, že téměř žádná klasifikační třída nepřesáhne hodnotu 50%. Jedním z možných příčin může být velká podobnost zpráv mezi jednotlivými klasifikačními třídami, což by naznačovalo, ze průnik dat není prázdný. Jinými slovy metoda nemůže najít dostatečně velké disjunktní shluky dat.

Tato nepřesnost mohla být do datasetu zanesena při počátečním třídění trénovacích dat v případě kdy zákazník položil dva odlišné dotazy v rámci jedné zprávy. Jedná se o problém, který v jisté míře ovlivňuje i ostatní modely, i když ne tak razantně.

Dev. test set	K=10 Freq	K=10 Tf-idf	K=5 Freq	K=5 Tf-idf	Množ.
Support	0.48	0.51	0.48	0.48	35
Registration_req.	0.58	0.62	0.58	0.59	38
Referer_id	0.52	0.44	0.16	0.12	13
Variable_req.	0.00	0.22	0.00	0.55	7
Support_req.	0.13	0.13	0.40	0.42	16
Module_instalation	0.60	0.67	0.63	0.67	50
Price_query	0.51	0.71	0.53	0.58	28
Foreign	0.29	0.00	0.12	0.14	10
Others	0.48	0.46	0.38	0.43	96
Customization_req.	0.00	0.00	0.15	0.14	11
Average	0.47	0.49	0.44	0.49	
Accuracy	135	**140**	128	136	304

Tabulka 6.3: **K-Neighbors - Base**. Každý řádek popisuje jednotlivé klasifikační třídy Každý sloupec definuje přesnost metody. **Freq**-použití frekvenčního slovníku během tréninku metody. **Tf-idf**-použití Tf-idf metriky během tréninku metody. **K**-počet sousedů potřebných pro zařazení do klasifikační třídy. **Vektor**-délka tréninkového vektoru byla stanovena na 100 vzorků. **Množ.**-množství testovacích dat v dané klasifikační třídě. **Average**-Průměrná přesnost. **Accuracy**-počet správně klasifikovaných zpráv.

6.4 Neuronová síť

Během implementace bylo experimentováno s různými neuronovými sítěmi s různými variacemi mohutnosti vstupních a skrytých vrstev, různým počtem skrytých vrstev. Všechny tyto sítě dosahovaly podobných výsledků, což byla tendence klasifikovat veškeré zprávy do majoritně zastoupené klasifikační třídy.

Eventuálně byla nalezena síť, jenž poskytovala použitelné výsledky. Výsledný tvar sítě je popsán v kapitole 4.5.1. Z tabulky 6.4 můžeme vidět, že se při použití krátkého trénovacího vektoru nestihne natrénovat. Daleko uspokojivějších výsledků dosahuje při použití delšího vektoru.

Získané výsledky z naši neuronové sítě jsou srovnatelné s metodou SVM, co se spolehlivosti týče. Samozřejmě je zde možnost, že se jedná pouze o shodu okolností.

Dev. test set	Freq(100)	Tf-idf(100)	Freq(6000)	Tf-idf(6000)	Množ.
Support	0.53	0.00	0.51	0.52	35
Registration_req.	0.57	0.32	0.70	0.64	38
Referer_id	0.00	0.00	0.87	0.80	13
Variable_req.	0.00	0.00	0.50	0.57	7
Support_req.	0.12	0.00	0.56	0.57	16
Module_instalation	0.60	0.48	0.72	0.68	50
Price_query	0.72	0.00	0.79	0.81	28
Foreign	0.00	0.00	0.67	0.75	10
Others	0.73	0.59	0.83	0.81	96
Customization_req.	0.00	0.00	0.00	0.00	11
Average	0.52	0.30	0.70	0.69	
Accuracy	177	128	**216**	215	304

Tabulka 6.4: **Neuronová síť - Base**. Každý řádek popisuje jednotlivé klasifikační třídy Každý sloupec definuje přesnost metody. **Freq**-použití frekvenčního slovníku během tréninku metody. **Tf-idf**-použití Tf-idf metriky během tréninku metody. **Vektor**-délka tréninkového vektoru (100)/(6000). **Množ**-určuje množství testovacích dat v dané klasifikační třídě. **Average**-průměrná přesnost. **Accuracy**-určuje počet správně klasifikovaných zpráv.

6.5 Modifikace vektoru

Výsledky předchozích testů byly zanalyzovány a na základě těchto výsledků byly navrženy modifikace jako odpověď na problémy s klasifikací. Tři klasifikátory ze čtyř klasifikují zprávy z našeho datasetu se spolehlivostí kolem 65%. Neznamená to tedy, že by natrénované modely pouze hádaly, ale ani to neznamená, že by přesně věděly, která zpráva patří kam. Provádí spíše informovaný odhad, který musíme nezbytně vylepšit. Ideálním řešením by bylo použití více tréninkových dat, což by vzhledem k manuálnímu třízení zpráv zabralo velké množství času. Z tohoto důvodu se pokusíme vylepšit data, která už máme.

Stojí za povšimnutí, že tréninková data tříd nejsou rovnoměrně zastoupena. Podíváme-li se na výsledky je jasně vidět, že nejpřesněji klasifikována třída je nejvíce zastoupená třída Others. Jak je možné vidět v confusion maticích C.1, C.2, C.6, C.7 třída Customization_request je velice málo zastoupena. Dodatečně je velice podobná jiným třídám, což vysvětluje tendenci těchto zpráv padnout do třídy Support.

Byly tedy provedeny modifikace při tvorbě vektoru, které by málo zastoupeným třídám zvýšily prioritu. První modifikací bylo vynásobení hodnoty vektoru počtem zpráv v dané třídě, abychom si ověřili hypotézu, že lze takto prioritu zvýšit 6.1.

$$\textbf{Mod1} = ClassCount(\text{"}Foreign\text{"}) * vector\_value \tag{6.1}$$

Druhá modifikace počítala s tím, že vynásobením počtem zpráv ve třídě zvýší všem třídám, proto byla navržena tak, aby vysoce zastoupeným třídám dávala menší prioritu a malým vetší. Hodnoty vektoru reprezentující daný dokument byly tedy vynásobeny hodnotou v rozsahu 1 až 10 dávaje vyšší číslo třídě méně zastoupené a menší třídě vysoce zastoupené 6.2. Druhá metoda ovšem počítá pouze s pořadím mohutnosti.

$$\textbf{Mod2} = rank\_of\_class(\text{"}Foreign\text{"}) * vector\_value \tag{6.2}$$

Třetí modifikace vynásobí hodnotu vektoru inverzní vahou dané třídy 6.3. Následně provedeme testy, jak tyto modifikace ovlivňují přesnost. Stejný test je proveden jak za použití hodnot frekvence slova tak hodnoty Tf-idf.

$$\textbf{Mod3} = Weight = \frac{Total}{ClassCount(\text{"}Foreign\text{"})} \tag{6.3}$$

6.5.1 Modifikace Naïve Bayes

Porovnáme-li hodnoty nemodifikovaného modelu 6.1 s nově naměřenými hodnotami po modifikacích 6.5 vidíme drobný nárůst v přesnosti. To by naznačovalo, že druhá a třetí modifikace funguje efektivněji nežli modifikace první.

Nicméně průměrný nárůst přesnosti je v rámci 4-5%, což může být způsobeno náhodnou odchylkou.

Dev. test set	F-Mod1	T-Mod1	F-Mod2	T-Mod2	F-Mod3	T-Mod3	Množ.
Support	0.42	0.41	0.48	0.49	0.49	0.46	35
Registration_req.	0.68	0.58	0.66	0.64	0.67	0.66	38
Referer_id	0.71	0.71	0.83	0.74	0.83	0.74	13
Variable_req.	0.53	0.35	0.56	0.53	0.50	0.56	7
Support_req.	0.35	0.47	0.47	0.44	0.53	0.51	16
Module_instalation	0.54	0.55	0.61	0.56	0.64	0.55	50
Price_query	0.65	0.58	0.72	0.66	0.78	0.68	28
Foreign	0.91	0.67	1.00	0.95	1.00	0.95	10
Others	0.77	0.76	0.78	0.76	0.79	0.76	96
Customization_req.	0.26	0.23	0.31	0.30	0.34	0.26	11
avg / total	0.63	0.60	0.67	0.64	0.69	0.64	304
Accuracy	187	180	200	193	**205**	193	

Tabulka 6.5: **Naïve Bayes/Mods**. Každý řádek popisuje jednotlivé klasifikační třídy Každý sloupec definuje přesnost metody. **F**-použití frekvenčního slovníku během tréninku metody. **T**-použití Tf-idf metriky během tréninku metody. **Mod#**-Jednotlivé modifikace v sekci 6.5. **Vektor**-délka tréninkového vektoru byla stanovena na 6000. **Množ**-určuje množství testovacích dat v dané klasifikační třídě. **Average**-průměrná přesnost. **Accuracy**-určuje počet správně klasifikovaných zpráv.

6.5.2 Modifikace SVM

Při opětovném porovnání výsledků modifikací modelu SVM v tabulkách 6.2 a 6.6 se jeví, že první modifikace přesnost naopak snižuje, zatímco zbylé dvě ji zvyšují.

Ve skutečnosti se přesnost stejně jako u předchozího modelu nijak výrazně nezlepšila.

Dev. test set	F-Mod1	T-Mod1	F-Mod2	T-Mod2	F-Mod3	T-Mod3	Množ.
Support	0.58	0.49	0.68	0.67	0.63	0.63	35
Registration_req.	0.56	0.53	0.66	0.65	0.66	0.66	38
Referer_id	0.60	0.56	0.70	0.75	0.76	0.82	13
Variable_req.	0.27	0.31	0.44	0.44	0.44	0.44	7
Support_req.	0.41	0.44	0.48	0.50	0.42	0.43	16
Module_instalation	0.55	0.61	0.60	0.57	0.61	0.60	50
Price_query	0.69	0.64	0.71	0.69	0 76	0.70	28
Foreign	0.35	0.35	0.53	0.46	0.43	0.33	10
Others	0.66	0.67	0.86	0.85	0.86	0.85	96
Customization_req.	0.27	0.30	0.40	0.12	0.44	0.13	11
avg / total	0.57	0.57	0.69	0.67	0.69	0.67	
Accuracy	171	170	215	210	**216**	211	304

Tabulka 6.6: **SVM/Mods**. Každý řádek popisuje jednotlivé klasifikační třídy Každý sloupec definuje přesnost metody. **F**-použití frekvenčního slovníku během tréninku metody. **T**-použití Tf-idf metriky během tréninku metody. **Mod#**-Jednotlivé modifikace v sekci 6.5. **Vektor**-délka tréninkového vektoru byla stanovena na 6000. **Množ**-určuje množství testovacích dat v dané klasifikační třídě. **Average**-průměrná přesnost. **Accuracy**-určuje počet správně klasifikovaných zpráv.

6.5.3 Modifikace K-nejbližších sousedů

Již víme, že metoda K-nejbližších sousedů na náš dataset není vhodná, nicméně i tak stojí za pokus se podívat, jak bude metoda ovlivněna. Při porovnání testovacího měření 6.3 s měřením modifikovaných vektorů 6.7 uvidíme, že všechny modifikace přispěly na přesnosti metody.

Co se týče první modifikace je rozdíl pouhých 4-5%, což není moc významné, ale co se týče ostatních modifikací je rozdíl kolem 9-10%.

Dev. test set	F-Mod1	T-Mod1	F-Mod2	T-Mod2	F-Mod3	T-Mod3	Množ.
Support	0.51	0.51	0.54	0.48	0.53	0.45	35
Registration_req.	0.70	0.69	0.62	0.59	0.64	0.62	38
Referer_id	0.23	0.19	0.60	0.64	0.57	0.61	13
Variable_req.	0.38	0.17	0.55	0.55	0.67	0.60	7
Support_req.	0.53	0.63	0.50	0.50	0.45	0.52	16
Module_instalation	0.60	0.66	0.53	0.62	0.56	0.65	50
Price_query	0.62	0.63	0.52	0.61	0.63	0.61	28
Foreign	0.33	0.33	0.29	0.35	0.40	0.40	10
Others	0.54	0.50	0.81	0.82	0.80	0.80	96
Customization_req.	0.27	0.29	0.15	0.00	0.00	0.00	11
avg / total	0.54	0.53	0.61	0.62	0.62	0.63	
Accuracy	153	149	194	**198**	197	**198**	304

Tabulka 6.7: **K-Neighbors/Mods**. Každý řádek popisuje jednotlivé klasifikační třídy Každý sloupec definuje přesnost metody. **F**-použití frekvenčního slovníku během tréninku metody. **T**-použití Tf-idf metriky během tréninku metody. **Mod#**-Jednotlivé modifikace v sekci 6.5. **K**-bylo stanoveno na 5 sousedů. **Vektor**-délka tréninkového vektoru byla stanovena na 100. **Množ**-určuje množství testovacích dat v dané klasifikační třídě. **Average**-průměrná přesnost. **Accuracy**-určuje počet správně klasifikovaných zpráv.

6.5.4 Modifikace neuronové sítě

Neuronová síť klasifikovala nejspolehlivěji vyjma tříd Variable_request a Customization_request. Ovšem porovnáme-li hodnoty v tabulce 6.4 vůči 6.8 vidíme již zaběhnutý trend. První modifikace negativně ovlivňuje neuronovou síť, zatímco zbylé dvě třídy metodu mírně vylepšují.

Opět se jedná o rozdíl několika procent, takže nemůžeme jednoznačně potvrdit, že se jedná o metody vylepšující přesnost.

Dev test set	F-Mod1	T-Mod1	F-Mod2	T-Mod2	F-Mod3	T-Mod3	Množ.
Support	0.47	0.51	0.60	0.60	0.54	0.56	35
Registration_request	0.68	0.70	0.70	0.74	0.72	0.73	38
Referer_id	0.78	0.83	0.83	0.87	0.83	0.83	13
Variable_req.	0.00	0.00	0.20	0.75	0.73	0.63	7
Support_req.	0.52	0.40	0.53	0.59	0.50	0.64	16
Module_instalation	0.67	0.63	0.67	0.70	0.70	0.68	50
Price_query	0.72	0.80	0.84	0.89	0.81	0.84	28
Foreign	0.75	0.84	0.89	1.00	0.89	1.00	10
Others	0.82	0.83	0.93	0.90	0.88	0.90	96
Customization_request	0.00	0.00	0.00	0.00	0.00	0.00	11
avg / total	0.66	0.69	0.73	0.76	0.73	0.75	
Accuracy	210	211	229	**237**	226	233	304

Tabulka 6.8: **Neural network/Mods**. Každý řádek popisuje jednotlivé klasifikační třídy Každý sloupec definuje přesnost metody. **F**-použití frekvenčního slovníku během tréninku metody. **T**-použití Tf-idf metriky během tréninku metody. **Mod#**-Jednotlivé modifikace v sekci 6.5. **Vektor**-délka tréninkového vektoru byla stanovena na 6000. **Množ**-určuje množství testovacích dat v dané klasifikační třídě. **Average**-průměrná přesnost. **Accuracy**-určuje počet správně klasifikovaných zpráv.

6.6 Shrnutí

Nyní když máme data před a po aplikaci námi navržených modifikací můžeme vidět drobná zlepšení přesnosti v rámci méně zastoupených klasifikačních tříd. Konkrétně se jedná o Variable_request a Customization_request a v případě K nejbližších sousedů i u třídy Others.

Kapitola 7

Vyhodnocení

Pro vypracování této práce bylo poskytnuto přes šest tisíc emailových zpráv z rozsahu pět let. Tyto zprávy jsou velmi různorodé. Jinými slovy se zprávy liší v délce, jazyku, gramatice, kódování, počtu témat v rámci jedné zprávy. V rámci této práce bylo roztřízeno 1496 zpráv, které byly použity k vytvoření slovníku a tréninku klasifikačních metod. Jelikož je množství dat poměrně malé, je zřejmé, že trénink nebude nejpřesnější. Tak či tak můžeme z nasbíraných testovacích dat získat obraz, jak bude probíhat budoucí vývoj.

7.1 Parser

Emailové zprávy přicházely v různých formátech a kódování z linuxového serveru společnosti TOPefekt.s.r.o. I přes všechny tyto problémy se podařilo vytvořit spolehlivý parser, který dokázal vyextrahovat relevantní obsah vyjma následujícího. V případě přeposlané zprávy bývá předchozí konverzace označena oddělovači, která umožňuje jejich snadné odstranění, viz 7.1.

Problém nastává v případech, kdy emailový klient, nebo odesilatel vloží předešlou konverzaci ve formě textu jako součást zprávy, viz 7.2. Jelikož tvar tohoto oddělení závisí na typu emailového klienta, nebo na formátování samotného uživatele není, jak rozpoznat kde končí relevantní zpráva a kde končí předešlá zpráva. Důsledkem je duplikace zpráv, která může, ale nemusí negativně ovlivnit kvalitu slovníku.

Ok sure, you can block sms to ********, we will do our best to
things like this will not ever happen again :)
thankyou, and let me know once you open account of ours :)

On Friday, October 30, 2015, Lubomír Kozák <kozak@topefekt.com>
wrote:

> Hi,
>
> according operator in ******** SMS were send to the number
**********.
>
> Operator in Pakistan has trouble with this and our company
also.
>
> You can check in your SMS history when SMS were send to
> particular number and it would be also good if you find the
> person who sent it and solve the problem on your side. Then
> we can enable the account for you again, but we must block SMS
> sending from your account to ******** or we will have problems.
>
> Best Regards

Obrázek 7.1: Předešlá konverzace (S oddělovači)

Hi Lunomir,

I think your SMS pack either infected or you inserted 64bit
codes. I found this on your latest version that's why I'm not
going to use your SMS service.
Sorry!

Regards,
*****

On Tuesday, August 5, 2014 6:09 PM, Lubomír Kozák
<kozak@topefekt.com> wrote:
Hi *****,

If on your credit account balance is displayed 103.100 credits
it means one hundred three point one credits.

Best Regards
Ing. Lubomir Kozak

Obrázek 7.2: Předešlá konverzace (Bez oddělovačů)

7.2 Slovník

Jak již bylo zmíněno, obdržené zprávy mohou být cizojazyčné, nebo mohou být plné pravopisných chyb. Tento fakt negativně ovlivňuje kvalitu slovníku. Slovník během své tvorby spočítá veškerá slova v přečteném obsahu a seřadí je nejdříve podle počtu výskytu a následně dle abecedy, což má za následek následující.

Každé slovo je převedeno na lower case slovo a posléze převedeno do kořenového tvaru. Tento proces snižuje výskyt výčtů stejných slov, které jsou uvedeny pouze v jiném tvaru. Nicméně pokud je slovo zapsáno gramaticky špatně, slovník si myslí, že se jedná o nové slovo, čímž vznikají nadbytečné záznamy ve slovníku. Tento problém by bylo možné odstranit kontrolou za pomoci dodatečných slovníků, ale použití pravopisného slovníku pro každý vyskytující se jazyk zvlášť, by razantně prodloužilo jeho tvorbu.

7.2.1 Cizojazyčnost

Původní hypotéza, že jazyk zprávy neovlivní klasifikaci, byl správný. Nicméně cizojazyčnost ovlivňuje délku slovníku a tím tedy i nepřímo trénink klasifikátoru. Jak již bylo zmíněno výše, slovník třídí dle počtu výskytů a posléze podle abecedy. Jelikož cizojazyčné zprávy se vyskytují méně než ostatní, výskyt cizojazyčných slov je velmi malý. Dodatečně cizí znakové sady jsou řazeny dle ascii tabulky až na konec slovníku.

Během tvorby trénovacího matice dochází k negativnímu jevu. Matice má dimenze NxM, kde N je počet unikátních slov ve slovníku a M je počet použitých zpráv. Chceme-li tedy efektivně natrénovat klasifikaci cizojazyčných zpráv, musíme zvolit takovou velikost slovníku, abychom obsáhli všechna relevantní slova. Tím pádem i zvýšili množství dat, což se nejvíce podepisuje na naší neuronové sítí. Pokud bychom se tedy rozhodli omezit slovník, mohlo by dojít k nechtěnému odstranění cizích slov a bez cizích slov je nemožné rozpoznat cizojazyčnou zprávu.

7.3 TF-IDF

Jelikož bylo nutné vytvořit vlastní parser pro čtení dokumentů, nebylo možné použít knihovních funkci pro výpočet Tf-idf. Testování ukázalo, že použití Tf-idf nezpůsobilo žádný razantní nárůst v přesnosti, což mohlo být způsobeno hned několika důvody. Jedním z těchto důvodů je, že ze zpráv byly vyfiltrovány všechna přebytečná slova, kterým by metoda Tf-idf přidělila menší skóre. Nejpravděpodobnějším důvodem je nerovnoměrné zastoupení tříd v datasetu. Může tomu být tak, protože metoda Tf-idf porovnává relevantnost termínu v dokumentu vůči celému datasetu. Nicméně některé termíny jsou unikátní v rámci klasifikační třídy. Dostanou tedy vyšší ohodnocení, jelikož se vyskytují často v rámci jedné klasifikační třídy, ale málo v rámci celého datasetu.

7.4 Klasifikační modely

Implementované modely byly natrénovány a otestovány za použití různě dlouhých tréninkových vektorových matic. Současně byly otestovány za použití rozdílných metrik pro hodnoty řečené matice. Slovník použitý k určení této matice byl vytvořen pouze za použití tréninkových dat. Tento slovník byl posléze taktéž použit ke stanovení hodnot v testovacím vektoru.

Z důvodu malého množství trénovacích dat, bylo rozdělení na trénovací a testovací data trochu upravena. Běžná praktika, je vzít všechna dostupná data a náhodně je rozdělit v po-

žadovaném poměru. V našem případě byla sesbírána všechna data jedné třídy a náhodně rozdělena dle požadovaného poměru. Tento postup byl použit pro rozdělení dat všech použitých klasifikačních tříd. Cílem bylo zařídit, aby vlivem náhodného generátoru čísel nedošlo k úplnému vypuštění jedné z menších klasifikačních tříd. Použití tohoto postupu mohlo ovlivnit objektivitu měření, ale rozdělení bylo stále dostatečně náhodné, že by to nemělo být problémem.

Všechny modely byly posléze otestovány za pomocí funkce Cross_val_score(), která provede cross validaci natrénováním specifikovaného modelu s rotačním rozdělením veškerých dat na trénovací a validační data, dle specifikovaného poměru. V našem případě byly testovány s poměrem 10% což vyžadovalo 10 iterací trénující a testující jeden model.

V tabulce 7.1 můžeme vidět průměrná hodnocení jednotlivých modelů za použití frekvenčních a Tf-idf hodnot trénovací vektorové matice. Model Naïve Bayes si v každé iteraci udržoval přesnost kolem 67% plus mínus 5%. Porovnávaje Naïve Bayes s modelem SVM dostáváme velice podobné výsledky, což naznačuje, že pro daný dataset emailových zpráv jsou tyto dva modely doposud nejvhodnější.

	NB-Fr	NB-T	SVM-Fr	SVM-T	K-Fr	K-T	NN-Fr	NN-T
1	0.70491	0.71311	0.66393	0.68852	0.43442	0.59016	0.	0.
2	0.61983	0.62809	0.70247	0.71900	0.41322	0.61157	0.15833	0.09166
3	0.60330	0.61983	0.65289	0.66115	0.51239	0.53719	0.19327	0.20168
4	0.70833	0.70833	0.74166	0.70833	0.45833	0.58333	0.25210	0.28571
5	0.69166	0.675	0.64166	0.68333	0.46666	0.625	0.41176	0.34453
6	0.64166	0.65833	0.68333	0.69166	0.35833	0.4	0.12605	0.13445
7	0.65833	0.65833	0.69166	0.7	0.55	0.59166	0.65546	0.65546
8	0.64655	0.68103	0.68103	0.65517	0.35344	0.53448	0.98319	0.97478
9	0.72413	0.71551	0.65517	0.6637	0.60344	0.60344	0.97478	0.98319
10	0.65517	0.70689	0.68103	0.68965	0.45689	0.56896	0.61344	0.61344
AVG	0.676449	0.676449	0.67948	0.686064	0.460717	0.564582	0.436841	0.428494
DEV	0.047335	0.046862	0.046923	0.047459	0.026827	0.035662	0.131676	0.133785

Tabulka 7.1: Cross validation Slouce tabulky udávají jednotlivé metody za použití frekvenčního slovníku (Fr) a metriky Tf-idf (T). Jednotlivé řádky udávají iterace trénování. Řádek AVG zobrazuje průměrnou přesnost F1 skóre skrze všechny iterace. Řádek DEV zobrazuje rozptyl přesnosti modelu.

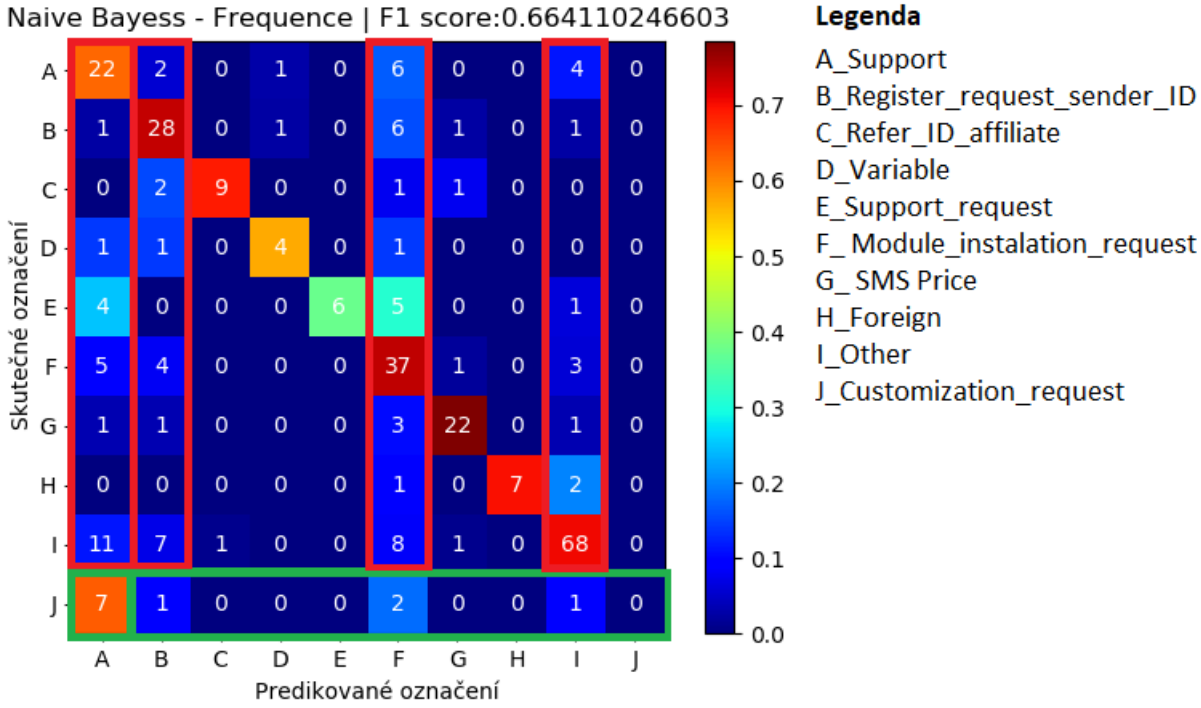
Dále zde máme model K-nejbližších sousedů a naše obavy byly potvrzeny. Model není pro náš dataset nejvhodnější. Možná by bylo možné s modifikací datasetu dosáhnout lepších výsledků, ale v tomto případě bude vhodnější použít jiný model. Nicméně v tabulce 7.1 vidíme, že model je stabilní nehldě na použitých datech.

Nakonec se dostáváme k vyhodnocení naši neuronové sítě a dosažená přesnost je uspokojující, bohužel různorodé hodnoty skrze iterace ukazují, že s největší pravděpodobností došlo k přetrénování modelu. Náš model je tedy velice závislý na použitých trénovacích datech. Posléze byly provázeny experimenty s konfigurací sítě, jako počet vnitřních vrstev, jejich velikosti, modifikace trénovací matice i modifikace slovníku, bohužel tyto modifikace nepřinesly žádné výrazné zvýšení přesnosti.

7.5 Případná vylepšení

Všechny problémy spojené s přesností modelů je hlavně způsobeny nedostatkem trénovacích dat, který je možné vyřešit, jelikož dodatečných dat je stále dostatek, nicméně je nezbytné jejich roztřízení.

Pokud se podíváme na výstup metody Naïve Bayes, viz obrázek 7.3. Zjistíme, že převážná většina nesprávně zařazených zpráv pochází ze čtyř klasifikačních tříd A-Support, B-Request_sender_ID, F-Module_Instalation_request, I-Other. V případě tříd A, B, F je hlavním problémem samotné třízení. Ve všech případech se jedná o žádost zákazníka o zprovoznění služby, její úpravu, nebo úpravy osobních údajů. Z toho můžeme usoudit, že modely dokáží rozpoznat požadavek zákazníka, ale nemají dostatek trénovacích dat, aby rozpoznaly jeden od druhého. Potencionálním řešením tohoto problému můžou být lépe roztřízená tréninková data, nebo sloučení některých klasifikačních tříd.



Obrázek 7.3: Zastoupení tříd

Třída I-Others obsahuje veškeré zprávy, které nesouvisí s ostatními třídami. Vysoká obecnost této třídy posléze způsobuje případné přiřazení zprávy do jiné třídy. Potencionálně by bylo vhodné třídu I-Other vůbec netrénovat a zařadit do třídy I-Other pouze ty zprávy, jejichž pravděpodobnost nepřesáhne prahovou hodnotu klasifikační třídy.

Nejhorších výsledků bylo dosaženo u třídy J_Customization_request. Stejně jako u tříd A, B, F se opět jedná o typ požadavku, nicméně tuto třídu je nejtěžší rozpoznat. Jednak je tomu tak, protože tato třída obsahuje malé množství trénovacích dat, ale zároveň zprávy žádající o modifikaci systému se obsahově velice podobají zprávám z tříd A, B, F žádající o modifikaci čehokoliv jiného. Nedostatečná přesnost této třídy je konzistentní skrze všechny použité metody a modely. Její odstranění a zařazení všech jejich zpráv do jiné třídy by jistě zvýšilo přesnost modelů.

7.5.1 Kombinace modelů

Stojí za povšimnutí, že některé třídy jsou klasifikovány lépe v závislosti na použitém modelu. Kupříkladu třídu Customization_request do určité míry rozpoznává model SVM, zatímco neuronová síť ji nerozpozná vůbec. Ale na druhou stranu neuronová síť rozpoznává třídu Registration_request mnohem lépe než ostatní metody.

Analyzovali jsme tedy získaná data a identifikovali, které modely jsou vhodnější pro specifické třídy. Posléze byla testovací data podrobena klasifikaci skrze všechny modely. Takto získaná data byla ve tvaru seznamu jednotlivých vektorů $\vec{A}, \vec{B}, \vec{C}, \vec{D}$, kde jednotlivé hodnoty udávaly pravděpodobnost s jakou si je model jistý zařazením zprávy do dané třídy. Následovně byly vytvořeny váhové vektory jednotlivých klasifikátorů $\vec{\alpha}, \vec{\beta}, \vec{\gamma}, \vec{\delta}$, kde hodnota na jednotlivých indexech reprezentovala spolehlivost klasifikace dané třídy. Tyto vektory byly použity pro výpočet lineární kombinace dle následujícího vzorce 7.1. Index nejvyšší hodnoty výsledného vektoru $\vec{x}$ udává výslednou klasifikační třídu.

$$\vec{x} = \sum_{i=1}^{k} \vec{\alpha_i}\vec{A_i} + \vec{\beta_i}\vec{B_i} + \vec{\gamma_i}\vec{C_i} + \vec{\delta_i}\vec{D_i} \tag{7.1}$$

Po provedení této modifikace se výsledná přesnost mírně zvýšila oproti ostatním modelů skrze většinu zastoupených tříd, viz tabulka 7.2.

Dev test set	NB	SVM	NN	Kombinace	Množ.
Support	0.51	0.63	0.54	0.56	35
Registration_request	0.67	0.66	0.72	0.74	38
Referer_id	0.78	0.76	0.83	0.83	13
Variable_request	0.62	0.44	0.73	0.67	7
Support_request	0.55	0.42	0.50	0.56	16
Module_instalation	0.62	0.61	0.70	0.67	50
Price_query	0.81	0.76	0.81	0.83	28
Foreign	0.82	0.43	0.89	1.00	10
Others	0.77	0.86	0.88	0.88	96
Customization_request	0.00	0.44	0.00	0.33	11
Average	0.66	0.69	0.73	0.75	
Accuracy	203	216	226	**229** - 75.32%	304

Tabulka 7.2: **Kombinace modelů** Řádky tabulky zobrazují jednotlivé klasifikační třídy. **NB/SVM/NN**-sloupec s výsledky metody Naïve Bayes, SVM a neuronové sítě. **Kombinace**-výsledky metody lineární kombinace. **Množ**-určuje počet testovacích dat v dané klasifikační třídě. **Average**-průměrná přesnost. **Accuracy**-určuje počet správně klasifikovaných zpráv.

7.6 Testování na reálném provozu

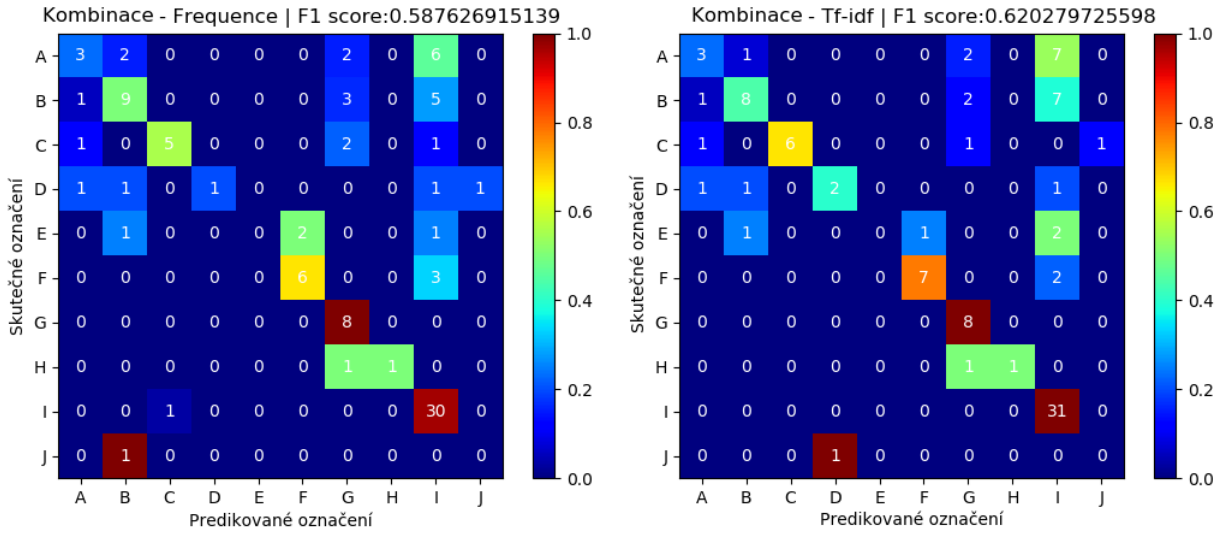
Všechny zprávy použité k natrénování modelů jsou poměrně staré. Nejstarší zpráva byla odeslána přibližně před pěti lety. Z tohoto důvodu byly provedeny testy, jenž měly ověřit funkcionalitu klasifikátoru na nových datech, abychom zjistili, zdali jsou klasifikátory vůbec použitelné. Byl tedy sestaven náhodně zvolený vzorek dat o velikosti **100** zpráv z nedávné komunikace se zákazníky.

Výsledky byly konzistentní s předešlými výsledky testů. Třídy, kterým byl poskytnut dostatek testovacích dat, dosahovaly nejvyšších přesností, viz tabulku 7.3. Dále se opětovně potvrdilo, že mnoho zpráv je si vzájemně velice podobných, viz confusion matice 7.4.

Taktéž je zde hypotéza, že klasifikátory fungují lépe, než se zdá a data byla z počátku pouze špatně roztřízena. Příkladem tohoto jevu je zpráva patřící do třídy J_Customization_request, ale byla klasifikována do třídy A_Support s pravděpodobností 0.999999820451. Z toho můžeme usoudit, že rozdíl mezi těmito dvěma třídami nebyl přesně definován.

Test set	Freq	Tf-idf	Množ.
Support	0.32	0.32	13
Registration_request	0.56	0.55	18
Referer_id	0.67	0.80	9
Variable_request	0.33	0.50	5
Support_request	0.00	0.00	4
Module_instalation	0.71	0.82	9
Price_query	0.67	0.73	8
Foreign	0.67	0.67	2
Others	0.77	0.77	31
Customization_request	0.00	0.00	1
Average	0.59	0.62	
Accuracy	**63** - 63%	**66** - 66%	100

Tabulka 7.3: **Klasifikace reálného provozu** Řádky tabulky zobrazují jednotlivé klasifikační třídy. **Váha**-sloupec určující, která metoda nejvíce přispívá ke klasifikační třídě. **Freq**-použití frekvenčního slovníku během tréninku metody. **Tf-idf**-použití Tf-idf metriky během tréninku metody. **Množ.**-určuje množství testovacích dat v dané klasifikační třídě. **Average**-Průměrná přesnost. **Accuracy**-určuje počet správně klasifikovaných zpráv.



Obrázek 7.4: Confusion matice klasifikace reálného provozu kombinační metodou. Testovací vektor levé matice byl vytvořen frekvenční metodou. Testovací vektor pravé matice byl vytvořen za pomocí Tf-idf. Hlavní diagonála matice určuje správně klasifikované zprávy jak bylo popsáno v kapitole 5.3

Kapitola 8

Závěr

V rámci této diplomové práce byla úspěšně vytvořena aplikace klasifikující emailovou komunikaci společnosti TOPefekt.s.r.o. Aplikace obsahuje čtyři metody klasifikace, které jsou běžně používané pro tyto účely. Vstupem aplikace jsou emailové zprávy z linuxového serveru řečené firmy a výstupem je výpis klasifikovaných zpráv společně s ohodnocením úspěšnosti. Ve stávající podobě není zprovozněna přímo na serveru, nicméně bude-li vyžadováno pokračovat v tomto projektu, budou podstoupeny úpravy pro splnění této funkcionality.

Součástí tohoto projektu bylo taktéž zjištění, zdali je klasifikace poskytnutých dat vůbec možná, jak se budou běžné metody chovat při použití takto komplikovaného datasetu a pokud možno modifikovat tyto metody pro vyšší přesnost. Pro dosažení vyšší přesnosti modelů bylo experimentováno s tvorbou slovníku, jeho velikostí, zápisem hodnot v tréninkové matici a interpretaci výsledků, nicméně z časových důvodů nebyl otestován vliv většího množství trénovacích dat na naše modely.

Obsah zpráv se pohyboval v rozsahu technické podpory, reklamy, žádosti o spolupráci, požadavků o instalaci, řešení nalezeného problému, nebo dotazy na cenu poskytovaných SMS. Všechny tyto zprávy se vyskytovaly v různých délkách jazycích, kódováních, formátech a četnosti. Pokud by to nebylo samo o sobě dost komplikované, všechna tato témata se mohla vyskytovat v rámci jedné zprávy.

Například zpráva obsahující firemní reklamu dotazující se na cenu SMS ve dvou různých měnách, zapsaná jak v ruštině, tak v rozbité angličtině. Tato zpráva byla následně několikrát přeposlána a nakonec na ni bylo několikrát odpovězeno. Přesně pro tyto případy byl v rámci této aplikace implementován spolehlivý parser, jenž pokryl všechny tyto možnosti, jak nejlépe to šlo.

Pro trénink těchto klasifikačních modelů bylo sesbíráno přes šest tisíc emailových zpráv z běžného provozu firmy. Během raných fázi projektu bylo roztřízeno kolem tisíce a půl emailových zpráv, které byly použity pro analýzu obsahu, rozdělení zpráv do klasifikačních tříd a následný trénink modelů. Posléze se ukázalo, že nejenom klasifikátory mají problémy s klasifikací. V mnoha případech se stane, že ani zaměstnanec firmy neví, kam zprávu zařadit, protože ani sám zákazník někdy neví přesně, co chce. Tyto nejasnosti se zákonitě podepíší i na našich modelech. Pokud tedy budeme chtít pokračovat v tomto projektu, bude zapotřebí, aby byly trénovací data rozdělena samotnými zaměstnanci firmy. Cílem je předejít nejasnostem mezi jednotlivými třídami. Ovšem tento proces by vyžadoval čas, kterým zaměstnanci nemohou mrhat.

I přes průměrnou přesnost 63% a problémy s klasifikací některých tříd můžeme říci, že by se klasifikátor mohl použít v reálném provozu. Je však nutné podotknout, že není možné ponechat třízení zpráv pouze na samotném klasifikátoru. Je možné jej využít jako

pomocný program zaměstnanci zodpovědného za třízení těchto zpráv. Vezmeme-li v potaz, množství použitých dat použitých při tréninku, tak je přesnost klasifikátoru stále velice dobrá. To nám ukazuje, že aplikování navržených úprav společně s větším množstvím dat má potenciál zvýšit přesnost klasifikace do požadovaných hodnot.

Literatura

[1] Bengfort, B.: *Text Classification with NLTK and Scikit-Learn.* [Online; navštíveno 11.01.2018].
URL https://bbengfort.github.io/tutorials/2016/05/19/text-classification-nltk-sckit-learn.html

[2] Britz, D.: *Deep Learning for Chatbots, Part 1 – Introduction.* [Online; navštíveno 11.01.2018].
URL http://www.wildml.com/2016/04/deep-learning-for-chatbots-part-1-introduction/

[3] Chien, S. W. J.-T.: *BayesianSpeechAndLanguageProcessing.* Cambridge University Press, 2015, ISBN 978-1-107-05557-5.

[4] Deshpande, A.: *Deep Learning Research Review Week 3: Natural Language Processing.* [Online; navštíveno 11.01.2018].
URL https://adeshpande3.github.io/adeshpande3.github.io/Deep-Learning-Research-Review-Week-3-Natural-Language-Processing

[5] Hastie T., F. J., Tibshirani R.: *The elements of statistical learning.* Springer-Verlag Berlin Heidelberg, 2009, ISBN 978-0-387-84858-7.

[6] Martinez W.L., S. J., Martinez A.R.: *Exploratory data analysis with MATLAB. 2nd edn.* Chapman & Hall/CRC, Boca Raton, 2011, ISBN 978-1439812204.

[7] Meloun M., M. J.: *Kompendium statistického zpracování dat.* Univerzita Karlova v Praze, 2013, ISBN 9788024621968.

[8] Neuveden: *Power laws, Pareto distributions and Zipf's law* . [Online; navštíveno 14.03.2018].
URL http://www-personal.umich.edu/~mejn/courses/2006/cmplxsys899/powerlaws.pdf

[9] Pedregosa, F.; Varoquaux, G.; Gramfort, A.; aj.: Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, ročník 12, 2011: s. 2825–2830.

[10] Wolfgang Karl Härdle, L. S.: *Applied multivariate statistical analysis.* Springer-Verlag Berlin Heidelberg, 2012, ISBN 978-3642172281.

Příloha A

Obsah DVD

- Zdrojové kódy klasifikátoru
- Text diplomové práce
- Manuál
- LaTeXové zdrojové kódy diplomové práce
- Prezentační video
- Plakát

Příloha B

Manuál

Pro spuštění klasifikátoru je nutné mít nainstalovaný Python 3.6 a následující knihovny B.1. Vstupem programu je adresářová struktura obsahující roztřízená data, viz B.2. Výstup klasifikace se vypisuje jak do konzole tak do výstupních souboru. Na konci běhu programu se taktéž vygenerují confusion matice z výsledků klasifikace.

B.1 Použité knihovny

- email-reply-parser 0.5.9
- numpy 1.13.3+mkl
- scikit-learn 0.19.0
- Keras 2.0.8
- nltk 3.2.5
- matplotlib 2.2.2

B.2 Vstupní adresářová struktura

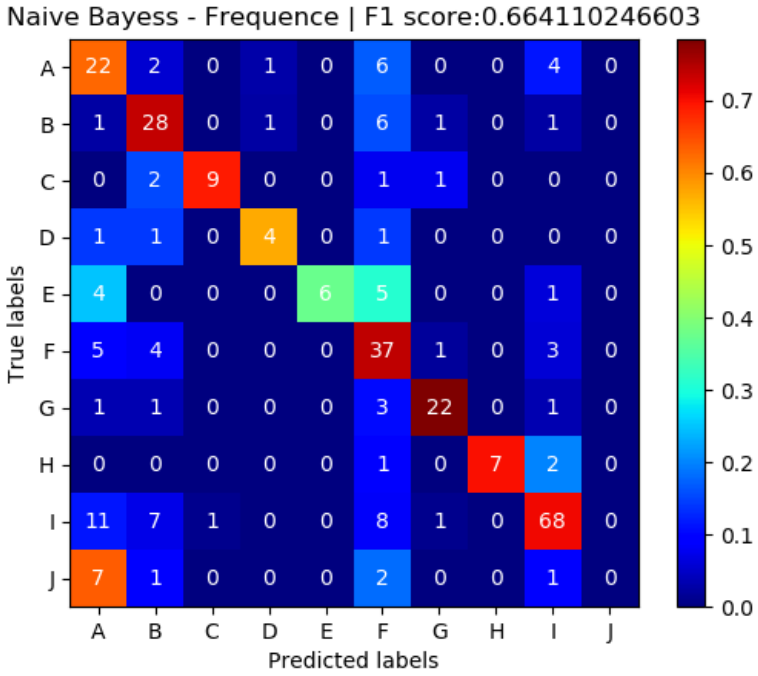
- a_type_support
- b_type_register_request_sender_ID
- c_type_referer_ID_affiliate
- d_type_variable
- e_type_support_request
- f_type_module_instalation_request
- g_type_SMS_price
- h_type_foreign
- i_type_other
- k_type_customization_request

B.3 Parametry programu

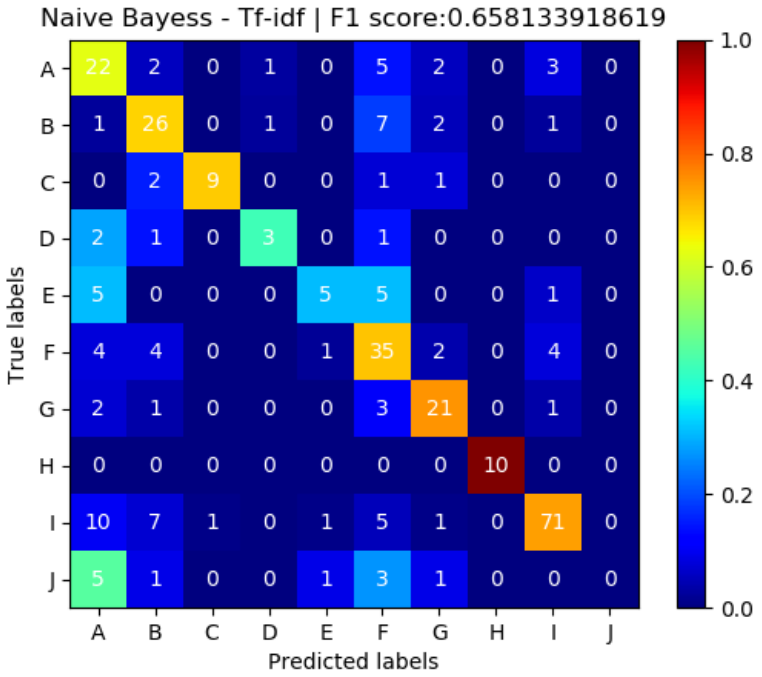
- classifier.py -i <directory> -m <method> -o <output>
- -h nápověda
- -i Vstupní adresář (Directory)
- -s Délka trénovacího vektoru
- -z Train and test split ratio (0.0 - 1.0)
- -t 0 - Vygeneruj klasifikační model / 1 - Použij již natrénovaný model
- -r 0 - Vygeneruj slovník / 1 - Použíj již natrénovaný slovník
- -m Klasifikační metoda (1 - NB / 2 - SVC / 3 - K-Nejbližších sousedů / 4 - Neuronová síť)
- -n Jméno modelu (<Jmeno_modelu> => NB_Tfidf_<Jmeno_modelu>)
- -d Metoda ohodnocení vektoru (0 - Frequency / 1 - TFIDF)

Příloha C

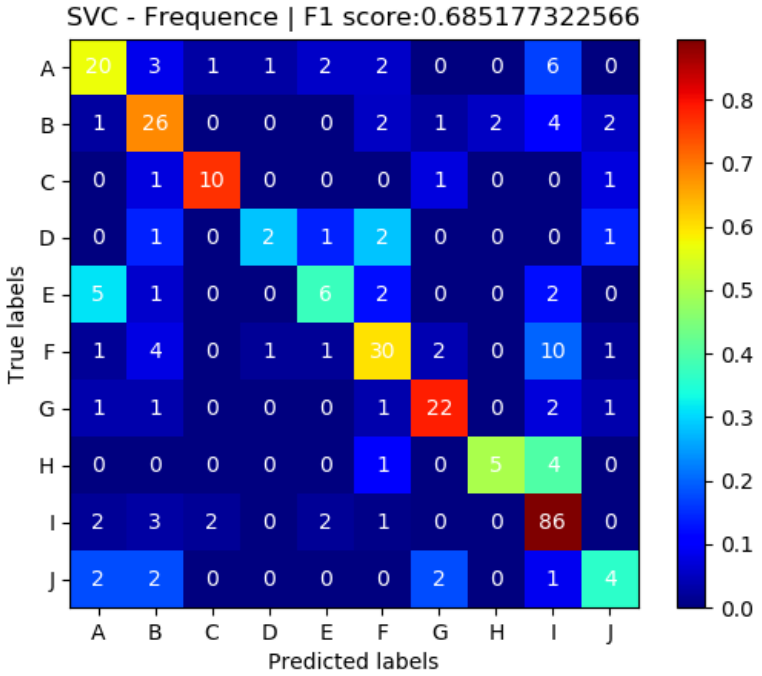
Confusion matice



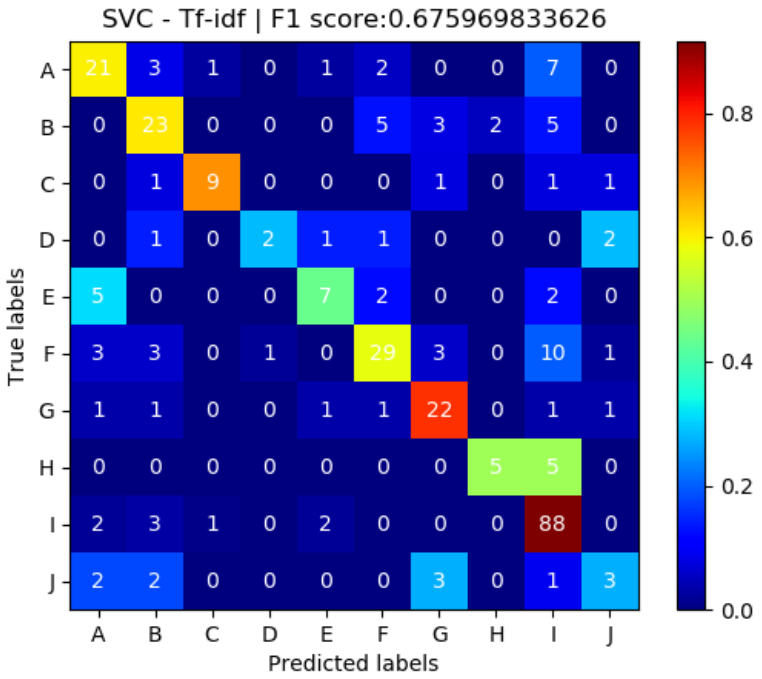
Obrázek C.1: Frequency - Naive Bayess



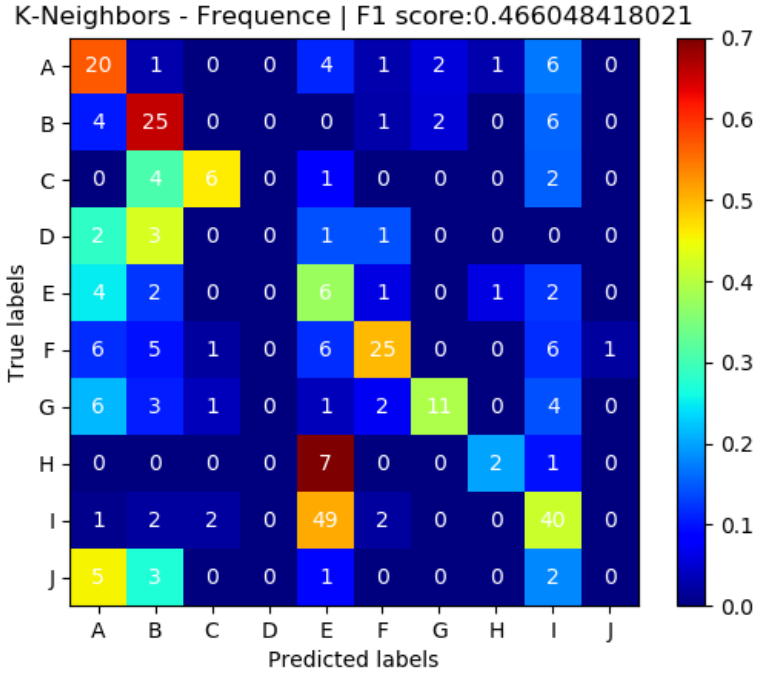
Obrázek C.2: Tf-idf - Naive Bayess



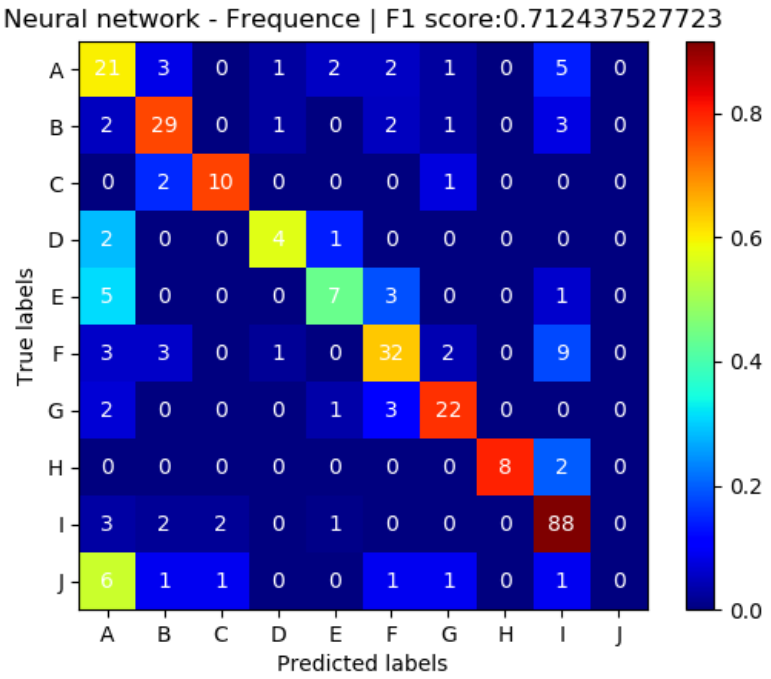
Obrázek C.3: Frequency - SVC



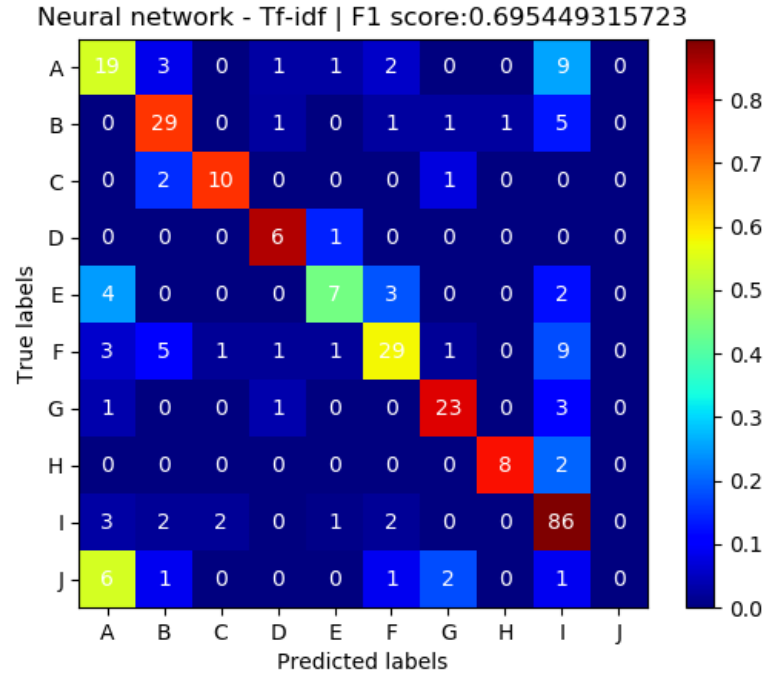
Obrázek C.4: Tf-idf - SVC



Obrázek C.5: Frequency - K-Neighbor



Obrázek C.6: Frequency - Neural network



Obrázek C.7: Tf-idf - Neural network