Vysoké učení technické

Fakulta informačních technologií

BIS - Bezpečnosť informačných systémov

Projekt 2

Spam filter

Martin Krajňák

xkrajn02

Úvod

Projekt je implementovaný v jazyku python3 za využitia knižnice na parsovanie správ eml_parse[1]. Detekcia spamu je riešená pomocou metódy založenej na učení za podpory knižnice scikit-learn[2]. Nasledujúce kapitoly obsahujú podrobnejší popis a riešené podproblémy.

Parsovanie správ

Na parsovanie hlavičiek správ a samotných správ je použitá už spomínaná knižnica eml_parse. Postupným testovaním bolo zistené, že aby došlo k správnemu rozoznaniu rôznych kódovaní a jazykov správ je vhodné použiť funkciu `eml_parser.decode_email_b(content, include_raw_body=True)` pričom obsah súboru (parameter content) musel byť otvorený v móde `'rb'`. Iné prístupy viedli k vyvolaniu chyby `UnicodeDecodeError`.

Po testovaní tohto parsera na niektorých emailoch zo sady poskytnutej v zadaní som bolo zistené, že parser nevie v niektorých prípadoch správne sparsovať správu. Tento jav bol pravdepodobne spôsobený zlým formátovaním eml súboru a vyskytoval sa len v súboroch, ktoré boli označené ako spam. Keďže nie je možné určiť s úplnou istotou, či šlo o chybu pri exporte alebo je táto chyba vo formátovaní spôsobená práve faktom, že daná správa je spam, rozhodol som sa, že v prípade výskytu takejto správy sa bude správa parsovať alternatívnym spôsobom.

Alternatívny spôsob parsovania správ tvorí algoritmus, ktorý otvorí správu v móde `'r'`, prejde obsah správy riadok po riadku a ako telo správy uloží všetok obsah súboru od prvého výskytu nového riadku po koniec súboru. Opätovne sa vyskytli možné problémy s čítaním súboru s určitým kódovaním, v takomto prípade sa tento súbor otvorí s parametrom `encoding=latin-1`, čo by malo zabezpečiť tzv. *best-effort* opening aj za predpokladu nesprávnej reprezentácie niektorých znakov.

Klasifikácia správ

Osobne nemám so strojovým učením žiadne predchádzajúce skúsenosti, takže vedomosti spojené s touto tématikou som čerpal článkoch[3][4] a na už spomenutej stránke scikit-learn, kde je tutoriál[5] venovaný spracovaniu textu. Z týchto zdrojov sa mi podarilo získať slušnú databázu[6] správ v angličtine, ktoré už boli klasifikované na spam a ham.

Problémom bol ale spam v českom a slovenskom jazyku. Keďže len minorita mailov obsahovala údaj `content-language`, nebolo možné určiť jazyk správy z hlavičky. Rozhodol som sa teda nazbierať vzorky spamu

1 http://eml-parser.readthedocs.io/en/latest/api.html
2 http://scikit-learn.org/stable/index.html
3 http://zacstewart.com/2015/04/28/document-classification-with-scikit-learn.html
4 http://pythonforengineers.com/build-a-spam-filter/
5 http://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html
6 http://www2.aueb.gr/users/ion/data/enron-spam/

a validných emailov kombináciou môjho osobného a školského emailu, ktorý by mal obsahovať české aj slovenské slová. Problém ale nastal keď boli pridané české spamy, preto vo finálnom datasete sú umiestnené len československé emaily a spam som vynechal, čím som dosiahol o poznanie lepšie výsledky.

Klasifikácia správ prebieha v niekoľkých krokoch. Prvá fáza samozrejme predstavuje načítanie správy a umiestnenie správy do datovej šktruktúry slovník, v ktorej sa nachádzajú dve položky a to telo správy a klasifikácia (`True` pre spam, `False` pre ham). Správy sú ukladané do zoznamu, ktorý je pred zahájením učenia premiešaný aby bol dosiahnutý lepší výsledok. Premiešanie je zabezpečované functiou shuffle a je veľmi dôležitou súčasťou, keď spam a ham sú uložené pokope.

Ďalšia fáza predstavuje vytvorenie inštancie objektu `CountVectorizer`, ktorý rozbije texty správ na jednotlivé slová a prevedie ich do matice, ktorá obsahuje počty výskytov jednotlivých slov. Otestovaná bola taktiež možnosť počítania dvojslov pomocou parametra `ngram_range`. Testovanie v tomtoprípade ukázalo, že výsledky nie sú oveľa lepšie, dokonca sa sa zvýšil počet správ falošne označených ako spam.

Samotné učenie sa prebiehá po vytvorení inštancia objektu `MultinomialDB`[7], ktorý pracuje na princípe naivného Bayesovho teorému, ktorý funguje na princípe podmienenj pravdepodobnosti. Princíp sa spolieha nato, že všetky vlastnosti (v tomto prípade slová) sú na sebe nezávislé a nezáleží natom, či sa určitá vlastnosť nachádza alebo nenachádza v dokumente. Všetky tieto vlastnosti sú vyhodnotené a na konci je spočítaná výsledná pravdepodobnosť, že sa objekt (v tomto prípade správa) radí buď do jednej alebo druhej triedy(spam alebo ham).

Na zjednodušenie práce spojenej s učením ponúka knižnica scikit-learn object `Pipeline`, ktorý zjednodušuje vytvorenie klasifikátora. Celý tento objekt je po skončení procesu učenia uložený a môžeme ho použiť na klasifikáciu nových správ. Uloženie priblieha pomocou volania metódy `joblib.dump()` z knižnice sklearn a podobne volaním `joblib.load()` je načítaný klasifikátor zo súboru.

V programe je časť, ktorá sa venuje učeniu pre istotu zakomentovaná, po spustení je načítaný klasifikátor zo súboru a prebieha klasifikácia. V tomto prípade je email sprasovaný a nad telom správy prebehne klasifikácia pomocou funcie `pipeline.predict([msg])`, ktorá prebehne správu a vypočita prevdepodobnostný vektor na základe ktorého je správa označená ako spam alebo ham. Výsledok je vypísaný v predpísanom formáte na štandardný výstup. Ak je programu predložený prečinok miesto zoznamu súborov alebo sa vyskytne akákoľvek iná chyba, program končí hlásením FAIL a vypísaním vstupu.

7 http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html#sklearn.naive_bayes.MultinomialNB