Primjena Bloom-ovog filtera u algoritmima detekcije noviteta

Damir Delijic 1/21 C

Contents

[Definicija problema 3](#_Toc137207597)

[Detekcija noviteta 3](#_Toc137207598)

[Bloom-ov filter 3](#_Toc137207599)

[Tekstualni dokumenti 3](#_Toc137207600)

[Algoritam 4](#_Toc137207601)

[Poredjenje sa drugim metodama 6](#_Toc137207602)

[Privremeno 6](#_Toc137207603)

[Literatura 6](#_Toc137207604)

# Definicija problema

Zadatak se sastoji u dizajniranju I implementaciji algoritma za detekciju noviteta u skupu tekstualnih dokumenata. Algoritam treba da bude zasnovan na odgovarajucoj modifikaciji Bloom-ovog filtera.

# Detekcija noviteta

Veliki broj aplikacija zahtijeva da postoji mogucnost odredjivanja da li nova obzervacija pripada istoj distribuciji kao I postojeci skup obzervacija, ili je treba kategorisati kao drugaciju.

Detekcija noviteta se smatra polu-nadgledanom metodom detekcije anomalija. Trening podaci se sastoje iz iskljucivo cistih obzervacija koje ne sadrze anomalije. Sistem bi nakon treninga trebao da ima sposobnost davanja odgovora na pitanje “Da li je nova obzervacija drugacija od poznatih?” sa odredjenom sigurnoscu.

# Bloom-ov filter

Bloom-ov filter je prostorno efikasna probabilisticka struktura podataka koja se koristi u svrhe testiranja pripadnosti elementa nekom skupu. Lazno pozitivni rezultati su moguci, dok lazno negativni nisu. Ideja je da se elementi mapiraju u koristeci neku hes funkciju , I onda testirati pripadnost provjeravanjem da li vazi

Prazni Bloom-ov filter je niz od m bita postavljenih na 0. Takodje mora se definisati k razlicith hes funkcija, od kojih svaka hesira element skupa u jednu od m pozicija niza bitova, generisuci unformnu nasumicnu distribuciju. Tipicno je k mala konstanta koja zavisi od zeljene preciznosti. Velicina niza bita, m je proporcionalna broju k I broju elemenata koji se dodaju.

Dodavanje elementa se vrsi tako sto se on predaje svakoj od k hes funkcija da bi se dobile k pozicija u nizu. Zatim se sve te pozicije setuju na 1.

Provjera da li neki element pripada se vrsi tako sto se preda svakoj od k hes funkcija I provjerava se da li su sve od dobijenih pozicija setovane na 1. Ako jesu onda element pripada skupu, a ako barem jedna nije onda ne pripada.

# Tekstualni dokumenti

Reprezentaciji tekstualnih dokumenata se moze pristupiti na vise nacina. Mogu se predstaviti kao skupovi rijeci koje se pojavljuju, kao skupovi kljucnih rijeci ili kako skupovi singlova. Jedino pristup sa singlovima uzima u obzir redoslijed rijeci.

Definicija k-singla je sekvenca od k tokena koji se pojavljuju u dokumentu. Tokeni mogu biti karakteri, rijeci ili nesto trece. Izbor tokena zavisi od primjene do primjene.

Dakle dokument je skup singlova, predstavljen kao nula-jedan vektor. Svaki singl je dimenzija I vektori su rijetki. Nad dokumentima se definise Jaccard-ova mjera slicnosti . su vektorske reprezentacije dokumenata uz pomoc singlova .

Odnosno . Gdje su:

# Algoritam

S obzirom da detekcija noviteta ukljucuje statisticku analizu kojom se odredjuje pripadnost nekoj distribuciji podataka, nije moguce koristiti Bloom-ov filter sa bilo kakvom hes funkcijom jer bi se na takav nacin moglo provjeriti samo da li je dati element dio trenutnog skupa podataka. Da bi se moglo provjeriti da li je element stvarni novitet, odnosno da nije jedan od vec postojecih I nije dovoljno slican ostalima da bi pripadao poznatoj distribuciji, neophodno je koristiti hes funkciju koja cuva odredjene osobine originalnih podataka, kao na primjer njihovu slicnost.

Dokumenti I njihova medjusobna slicnost su definisani kao u prethodnom naslovu. Treba iskoristiti hes funkciju h takvu da ako je slicnost dokumenata velika, onda je velika I vjerovatnoca da su im hes vrijednosti identicne. Jedna takva hes funkcija koja je definisana za Jaccard-ovu slicnost je Minhashing.

Dokumenti su predstavljeni pomocu matrice tako sto su redovi prestavljaju singlove, a kolone predstavljaju dokumente. Za i-tu kolonu ako je j-ti red jedinica znaci da j-ti singl pripada i-tom dokumentu.

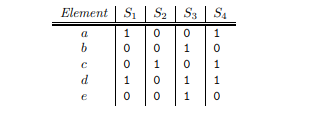


Figure : Reprezetacija dokumenata pomocu karakteristicne matrice

Hes vrijednost dokumenta te matrice je pozicija prvog reda u kom se nalazi jedinica za taj dokument.

Na primjer u slici 1. hes vrijednosti dokumenata su redom .

Veza izmedju Jaccard-ove slicnosti I minhasing-a je ta da je vjerovatnoca da je hes vrijednost dva dokumenta nakon nasumicne permutacije redova ista jednaka njihovoj Jaccard-ovoj slicnosti. Dokaz se moze naci u knjizi [3] poglavlje 3.3.3.

Sto bi trebalo da znaci da kada se poznati dokument hesira minhashing metodom I smjesti u Bloom-ov filter, svi dokumenti koji su mu veoma slicni po Jaccard-u bi trebali da imaju istu hes vrijednost.

Ovom metodom se omogucava da pozicija u filteru “prima” vise vrijednosti koje se nalaze u blizini stvarne tacke cime se olaksava uslov I postaje moguce detektovati elemente koji su prilicno slicni stvarnim umjesto samo one koji su identicni nekom od stvarnih.

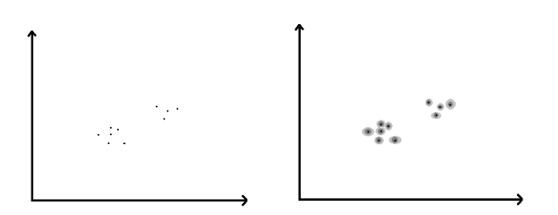


Figure Standardni pristup lijevo, minhashing pristup desno. Minhashing cuva udaljenost dokumenata

Nije izvodljivo permutovanje velike karakteristicne matrice eksplicitno. Cak je I nasumicno biranje permutacije miliona redova dugotrajno, a sortiranje takvih redova jos dugotrajnije. Pa, tako permutovane matrice, iako su konceptualno primamljive, nije moguce implementirati.

Srecom, moguce je simulirati efekat nasumicne permutacije pomocu hes funckije koja mapira redne brojeve redova matrice u onoliko bucket-a koliko ima redova. Takva funkcija ce tipicno mapirati neke parove brojeva u iste korpe a ostale ostaviti praznim. Razlika je nebitna dokle god je broj redova velik I nema puno kolizija. Hes funkcija h “permutuje” red r u poziciju h(r) u permutovanom redoslijedu.

Pa umjesto biranja n nasumicnih permutacija redova, biramo n nasumicnih hes funkcija h1, h2, … , hn redova. Konstruisemo matricu signatura kretanjem po redovima od pocetka. Neka je Sig(I, c) element matrice signature za i-tu hes funkciju I kolonu c. Inicijalno su svi elementi matrice singnatura podeseni na najvecu mogucu vrijednost.

Implementacija algoritma se sastoji iz sledeceg:

* Izabere se broj koji predstavlja broj bita memorije koji su dostupni.
* Na osnovu njega I broja dokumenata se racuna optimalno tako da procenat lazno pozitivnih bude nizak.
* Inicijalizuje se prazan Bloom filter.
* Nasumicno se generisu k prirodnih brojeva da bi sluzili kao seed hes funkciji.
* Inicijalizuje se matrica signatura.
* Na osnovu dostupnih podataka se formira matrica signatura na nacin kako je opisano u knjizi [3] u poglavlju 3.3.5.
* Rezultujuci filter zajedno sa parameterima potrebnim za detekciju se cuvaju u fajlu

Za tesitranje I laksu implementaciju, pisana je skripta *data\_generator.py* koja generse matricu bita sa zadatim dimenzijama I elemente stavlja na nulu ili jedinicu sa zadatom vjerovatnocom.

# Poredjenje sa drugim metodama

# Privremeno

# Literatura

[1] <https://scikit-learn.org/stable/modules/outlier_detection.html>

[2] <https://en.wikipedia.org/wiki/Bloom_filter>

[3] http://infolab.stanford.edu/~ullman/mmds/book.pdf