Klasifikacija tekstualnih dokumenata

Cilj ove vježbe je upoznavanje sa različitim algoritmima za klasifikaciju tekstualnih dokumenata. Kao primjer zadatka klasifikacije iskoristićemo detekciju spam email poruka. Dakle, potrebno je projektovati binarni klasifikator kod kojeg su klase u koje se uzorci klasifikuju spam i ham. U vježbi će se koristiti tri klasifikatora: k najbližih susjeda (kNN), metod vektora nosača (SVM) i naivni Bejsov (NB) klasifikator.

1 NB klasifikator

NB klasifikator svrstava dokument u klasu kojoj odgovara maksimalna a posteriori (MAP) vjerovatnoća da dokument d pripada klasi c

$$\hat{c} = \arg\max_{c \in C} P\left(c \mid d\right). \tag{1}$$

Na osnovu Bejsovog teorema a posteriori vjerovatnoća se može izračunati kao

$$P\left(c \mid d\right) = \frac{P\left(c\right) P\left(d \mid c\right)}{P\left(d\right)}. \tag{2}$$

Pošto MAP klasifikator odluku donosi na osnovu maksimuma a posteriori vjerovatnoće, nije potrebno izračunavati $P\left(d\right)$ jer ne zavisi od oznake klase. Dakle, problem se svodi na određivanje

$$\hat{c} = \arg\max_{c \in C} P\left(c\right) P\left(d \mid c\right). \tag{3}$$

Osnovna pretpostavka NB klasifikatora je da su termini uslovno nezavisni, tj. da su, za datu klasu, termini međusobno nezavisni, te da uslovna vjerovatnoća pojavljivanja termina u klasi c ne zavisi od pozicije u dokumentu. Sada se uslov (3) može napisati u obliku

$$\hat{c} = \arg\max_{c \in C} \hat{P}\left(c \mid d\right) = \arg\max_{c \in C} \hat{P}\left(c\right) \prod_{1 \leq k \leq n_d} \hat{P}\left(t_k \mid c\right), \tag{4}$$

gdje je n_d broj termina u dokumentu d, a oznaka $\hat{P}$ se koristi da bi se naglasilo da su vrijednosti vjerovatnoća estimirane na osnovu primjera iz trening skupa. Pošto proizvod u jednačini (4) može poprimiti veoma male vrijednosti, da bi se izbjegli numerički problemi umjesto vjerovatnoća se koriste njihovi logaritmi. Pošto je logaritamska funkcija monotona, dokument se klasifikuje u klasu koja odgovara maksimalnoj vrijednosti logaritma vjerovatnoće

$$\hat{c} = \arg\max_{c \in C} \log \hat{P}(c \mid d) = \arg\max_{c \in C} \left[\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} log\hat{P}(t_k \mid c) \right]. \quad (5)$$

A priori vjerovatnoće klasa se mogu estimirati jednačinom

$$\hat{P}(c) = \frac{N_c}{N}, \quad (6)$$

gdje je N_c broj trening dokumenata koji pripadaju klasi c, a N ukupan broj trening dokumenata. Pri estimaciji uslovnih vjerovatnoća $P(t \mid c)$ moguće je pretpostaviti jedan od dva modela generisanja dokumenta iz date klase.

Kada se pretpostavi *Bernulijev model* estimacija $\hat{P}(t \mid c)$ se računa kao dio dokumenata iz klase c koji sadrže termin t. Dakle, broj pojavljivanja termina nije bitan već samo da li se on pojavljuje u dokumentima iz klase c

$$\hat{P}(t \mid c) = \frac{N_{ct} + 1}{N_c + 2}, \quad (7)$$

gdje je N_{ct} broj trening dokumenata iz klase c koji sadrže termin t, a N_c ukupan broj trening dokumenata iz klase c. U jednačini (7) konstante su dodate da bi se izbjegla situacija u kojoj se neki termin ne javlja u dokumentima iz trening skupa što bi rezultovalo nultom uslovnom vjerovatnoćom pojavljivanja tog termina.

Sa druge strane, *multinomni model* estimira $\hat{P}(t \mid c)$ kao odnos broja pojavljivanja termina t u dokumentima iz klase c i ukupnog broja pojavljivanja svih termina u dokumentima iz klase c

$$\hat{P}(t \mid c) = \frac{T_{ct} + 1}{\sum_{t' \in V} (T_{ct'} + 1)} = \frac{T_{ct} + 1}{\sum_{t' \in V} T_{ct'} + B} \quad (8)$$

gdje je T_{ct} broj pojavljivanja termina t u trening dokumentima iz klase c, uključujući i višestruka pojavljivanja, a B je broj elemenata rječnika V. U jednačini (8) konstante su dodate da bi se izbjegla situacija u kojoj se neki termin ne javlja u dokumentima iz trening skupa što bi rezultovalo nultom uslovnom vjerovatnoćom pojavljivanja tog termina.

2 Uputstva za praktični rad

2.1 LIBSVM biblioteka

Kao jedan od klasifikatora u ovoj vježbi biće korišćene i SVM. SVM biblioteku LIBSVM je moguće preuzeti sa `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`. MATLAB interfejs u aktuelnoj verziji je kompajliran za 64-bitne procesore. Ukoliko koristite 32-bitni procesor potrebno je preuzeti stariju verziju biblioteke sa `http://www.csie.ntu.edu.tw/~cjlin/libsvm/oldfiles/libsvm-3.12.zip`[1]. Preuzetu ZIP arhivu je potrebno raspakovati na pogodno mjesto, a zatim u MATLAB-ovu listu putanja (**File** > **Set Path**) dodati putanju do direktorijuma `%LIBSVM%\windows`, gdje je `%LIBSVM%` direktorijum u koji je raspakovana LIBSVM arhiva. Nakon ovoga biblioteka je spremna za korišćenje iz MATLAB-a. Uputstvo za pozivanje funkcija iz LIBSVM biblioteke iz komandne linije nalazi se u fajlu `README` u direktorijumu `%LIBSVM%\`, a uputstvo za pozivanje funkcija iz MATLAB-a u fajlu `README` u direktorijumu `%LIBSVM%\matlab`.

Osnovne funkcije iz LIBSVM biblioteke koje ćemo koristiti u ovoj vježbi su `svmtrain` i `svmpredict`. Pomoću `svmtrain` se obučava SVM klasifikator. Sintaksa funkcije je

```
model = svmtrain(training_label_vector, training_instance_matrix
                 [, 'libsvm_options']);
```

- `training_label_vector` je vektor dimenzija $m \times 1$ koji sadrži oznake primjera iz trening skupa. Tip ovog vektora mora biti `double`.
- `training_instance_matrix` je matrica dimenzija $m \times n$ koja sadrži m instanci trening uzoraka sa po n obilježja. Tip ove matrice mora biti `double`.
- `libsvm_options` je string koji sadrži opcije algoritma za obučavanje kao što je opisano u `%LIBSVM%\README`. Neke od korisnih opcija su:
  - `-t tip_kernela` – izbor tipa kernela (predefinisano 2):
    - 0 – linearni: $u^T v$,
    - 1 – polinomski: $\left(\gamma u^T v + c_0\right)^{red}$,
    - 2 – radijalna bazna funkcija: $\exp\left(-\gamma \|u - v\|^2\right)$
    - 3 – sigmoidni: $\tanh\left(\gamma u^T v + c_0\right)$

[1] Verzija 3.12 je poslednja koja ima kompajliran MATLAB interfejs za 32-bitne procesore

    - 4 – unaprijed izračunati kernel (vrijednosti u zadatom fajlu)
  - `-d red` – red kernel funkcije za polinomski kernel (predefinisano 3)
  - `-g gama` – vrijednost γ za RBF kernel (predefinisano 1/broj_obilježja)
  - `-c parametar_C` – vrijednost parametra C

Funkcija `svmtrain` vraća model koji se može koristiti za predikciju klasa kojim pripadaju novi uzorci. Polja ove strukture opisana su u dokumentaciji.

Pomoću funkcije `svmpredict` određuje se predikcija klase kojoj pripada novi uzorak. Sintaksa ove funkcije je

```
[predicted_label, accuracy, decision_values/prob_estimates] =
          svmpredict(testing_label_vector, testing_instance_matrix,
          model [, 'libsvm_options']);
```

- `testing_label_vector` je vektor dimenzija $m \times 1$ koji sadrži oznake testnih primjera. Ako oznake testnih primjera nisu poznate moguće je koristiti proizvoljne vrijednosti. Tip ovog vektora mora biti `double`.
- `testing_instance_matrix` je matrica dimenzija $m \times n$ koja sadrži m testnih primjera sa po n obilježja. Tip ove matrice mora biti `double`.
- `model` je struktura koju je vratio poziv `svmtrain`
- `libsvm_options` je string sa testnim opcijama koje se zadaju u istom formatu kao i kada se funkcija poziva iz komandne linije.

Prvi izlazni argument funkcije, `predicted_label`, je vektor predikcija oznaka klasa testnih primjera. Drugi izlazni argument, `accuracy`, je tačnost klasifikacije. Treći izlazni argument je vrijednost funkcije odlučivanja, odnosno, estimacije vjerovatnoća da uzorak pripada svakoj od klasa (ako se koristi opcija '`-b 1`').

2.2 Formiranje vektorske reprezentacije dokumenata

MATLAB kod za formiranje vektorske reprezentacije i primjeri email poruka dati su na web stranici predmeta `http://dsp.etfbl.net/pms/spam.zip`. U direktorijumu `emails` su email poruke podijeljene u trening i test skupove, te manuelno klasifikovane kao spam ili ham. U datim m-fajlovima implementirano je generisanje rječnika i formiranje vektorske reprezentacije za svaki dokument iz kolekcije.

Kod je organizovan na sljedeći način:

- `spam_classifier`
  Glavni program iz kojeg se pozivaju funkcije za generisanje rječnika i formiranje vektorske reprezentacije za svaki dokument iz kolekcije, te funkcije za obučavanje i testiranje klasifikatora;

- `conf = get_conf()`
  Funkcija koja vraća strukturu `conf` u kojoj su zadati parametri klasifikatora;

- `vocab = generate_vocabulary(conf)`
  Generiše rječnik za trening dokumente koji se nalaze na zadatoj putanji. Ulazni argument je struktura sa parametrima klasifikatora. Vraća ćelijski niz koji sadrži termine koji čine rječnik.

- `[training_set, training_C, test_set, test_C] = compute_collection_representation(vocab, conf)`
  Izračunava vektorske reprezentacije za sve dokumente iz kolekcije. Ulazni argumenti su rječnik i struktura sa parametrima klasifikatora. Vraća matrice `training_set` i `test_set` u kojima su redovi vektorske reprezentacije trening i test dokumenata, te vektore `training_C` i `test_C` koji sadrže oznake klasa za svaki trening, odnosno, test dokument. Vektorske reprezentacije dokumenata su normalizovane tako da je njihova L_2 norma jednaka jedinici.

- `vector = compute_document_representation(filepath, vocab, conf)`
  Izračunava vektorsku reprezentaciju za dokument u fajlu na zadatoj putanji. Ulazni argumenti su i rječnik te struktura sa parametrima. Funkcija vraća vektor čiji elementi su brojevi pojavljivanja pojedinih termina iz rječnika.

- `label = apply_knn(training_set, training_C, test_set)`
  Klasifikator na principu najbližih susjeda. Ulazni argumenti su matrica `training_set` dimenzija $m \times n$ koja sadrži m trening primjera sa po n obilježja, vektor `training_C` dimenzija $m \times 1$ koji sadrži oznake trening primjera i matrica `test_set` dimenzija $l \times n$ koja sadrži l testnih primjera dimenzionalnosti n. Izlazni argument je vektor predikcija oznaka testnih primjera.

- `Cm = conf_mat(ctest_hat, ctest, Nclasses)`
  Izračunava matricu konfuzija za dati vektor predikcija oznaka testnih primjera, `ctest_hat`, vektor tačnih oznaka testnih primjera, `ctest` i broj klasa. Izlaz iz funkcije je matrica konfuzija.

3 Zadatak

1. Upoznati se sa arhitekturom klasifikatora i osnovnim strukturama podataka.

2. Testirati klasifikator email poruka pokretanjem programa `spam_classifier`. Implementirani su SVM klasifikator i klasifikator na principu najbližih susjeda. Kolike su tačnosti svakog od klasifikatora?

3. Varirati minimalan broj pojavljivanja riječi u intervalu od 1 do 9 sa korakom 2 i minimalnu dužinu riječi od 1 do 5 sa korakom 2. Kako se mijenja tačnost klasifikacije?

4. Implementirati kNN klasifikator. Testirati klasfikator na datim primjerima.

5. Implementirati NB klasifikatore sa multinomnim i Bernulijevim modelima. Testirati klasifikator na datim primjerima. Varirati minimalan broj pojavljivanja riječi u intervalu od 1 do 9 sa korakom 2 i minimalnu dužinu riječi od 1 do 5 sa korakom 2. Kako se mijenja tačnost klasifikacije? Algoritam je dat na predavanjima, a sintaksa poziva funkcija je data u `spam_classifier`. Potrebno je napisati funkcije koje će estimirati a priori vjerovatnoće klasa $\hat{P}(c)$ i uslovne vjerovatnoće pojavljivanja svakog od termina u pojedinim klasama $\hat{P}(t \mid c)$, a zatim, za testni dokument, izračunavati ocjenu pripadnosti dokumenta klasama. Formirati matrice konfuzija ovih klasifikatora i izračunati tačnosti klasifikacije.