Pronalaženje skrivenog znanja
Projektni zadatak za školsku 2024/2025. godinu

Projektni zadatak se sastoji iz pet celina na kojima se može ostvariti ukupno 60 poena. Zadaci se odnose na prikupljanje podataka, njihovu analizu i obradu, vizuelizaciju i implementaciju modela mašinskog učenja korišćenjem algoritama mašinskog učenja, kao i evaluaciju modela. Obavezno je uraditi bar jedan zadatak iz skupa {4 i 5} da biste ostvarili prolazan broj poena.

Projektni zadatak se radi samostalno. Predati projekti se upoređuju međusobno. Studenti kod kojih se utvrdi sličnost dobijaju 0 poena na projektu u ovoj školskoj godini i u ukupnom broju poena (bez obzira da li su izlazili na pismeni ispit) i nemaju prava da izlaze više na odbrane projekata u ovoj školskoj godini. Stoga molimo studente da samostalno rade svoje projekte, kako ne bismo bili prinuđeni da prijave šaljemo Disciplinskoj komisiji.

Zadatak 1: Prikupljanje podataka

Realizovati veb indekser (eng. *web crawler/web spider*) sa veb parserom (eng. *web scraper*), koji prikuplja podatke o turističkim aranžmanima sa jednog ili više od sledećih sajtova:

- https://kontiki.rs/
- https://www.wayout.rs/sr
- https://www.rapsodytravel.rs/
- neki sajt turističke agencije u Srbiji koji nije u ovoj listi, a ima dovoljan broj zapisa.

Formirati sopstvenu relacionu bazu podataka sa svim relevantnim informacijama o aranžmanima koji se nude. Bazu realizovati kao relacionu, u tehnologiji *MySQL* ili *PostgreSQL*. Baza treba da ima najmanje 8 hiljada aktuelnih zapisa o aranžmanima.

Šta je veb indekser?

Cilj veb indeksera je da se poveže na određenu veb stranu i da preuzme njen sadržaj. Parsiranjem date strane možemo naći linkove, koji vode na neke druge strane, na koje veb-indekser ponovo može da uđe i da ponovi celu proceduru. Pored otkrivanja linkova, parser može da prepozna i druge sadržaje koje veb strana ima. U vašu bazu treba da prikupite informacije o svim aranžmanima – naziv hotela, mesto, broj zvezdica hotela, tip usluge, broj noćenja, datum polaska i cena po osobi. Podaci koji nisu dostupni u opisu, u bazi treba da ostanu praznog polja.

Implementaciju veb-indeksera možete raditi u programskim jezicima: C, C++, C#, Java, Python, NodeJS ili PHP. Dozvoljeno je i korišćenje i prilagođavanje neke od postojećih implementacija otvorenog koda: *crawler4j*, *Heritrix*, *Nutch*, *Scrapy* za Python, *PHP-Crawler* za PHP, itd.

Zbog ograničenog broja zahteva na serverima sa iste IP adrese, koristiti rotirajuće proxy-je ili neku drugu tehniku, kako ne biste kršili uslove korišćenja usluga izvornih veb sajtova. Prikupljanje ovih podataka koji nisu orijentisani ka ličnim podacima i koji jesu javno dostupni je dozvoljeno, ali uz molbu da broj zahteva ka serveru prilagodite, i da između zahteva bude vremenske razlike, kako ne biste domaćinu sajta izvršili *Denial-of-service attack (DoS)*.

Šta je veb parser?

Uloga veb parsera je da otkrije potreban sadržaj sa primljenih veb strana. Pri tome potrebno je odrediti značenje sadržaja kako bi se baza podataka popunjavala tačnim podacima. Najčešće tehnike koje se koriste pri implementaciji veb parsera su: HTML parser, DOM parser, tehnika regularnih izraza koji izdvajaju potreban sadržaj i tehnika prepoznavanja semantičkih anotacija. Za potrebe veb parsiranja takođe možete koristiti neku od postojećih implementacija (npr. biblioteka *Jsoup* – parsira veb stranu kao stablo elemenata, *BeautifulSoap*, *Scrapy*, ili *PySpyder*, za Python, itd.).

Kao rezultat zadatka 1 treba da prikažete realizovanu relacionu bazu podataka popunjenu traženim podacima o aranžmanima i da priložite implementacije koje su korišćene za dohvatanje podataka. Podaci treba da budu preuzeti u konačnom vremenskom intervalu.

Zadatak 2: Analiza podataka

Iz navedenih zapisa ubačenih u bazu podataka (iz zadatka 1), potrebno je preprocesirati podatke (odbaciti one koji nemaju veći broj potrebnih vrednosti polja/dopuniti neka polja), zabeležiti to u novoj bazi (sa brojem zapisa koji je preostao, ne manji od 7 hiljada), i uraditi sledeće:

a) izlistati koliki je broj aranžmana po mestima;
b) izlistati koliko aranžmana se realizuje u hotelima sa četiri i pet zvezdica;
c) izlistati koliko aranžmana nudi uslugu „all inclusive“;
d) izlistati koliko aranžmana nudi opcije za 12 i više noćenja;
e) prikazati rang listu prvih 30 najskupljih aranžmana;
f) prikazati aranžmane (Top30) koji imaju:
    - uslugu „all inclusive“,
    - najveću cenu.

Kao rezultat zadatka 2 treba priložiti bazu podataka (revidiranu i prečišćenu, iz zadatka 2), realizovane upite i generisane rezultate (npr. izvesti u *Excel* ili *Word* fajl).

Zadatak 3: Vizuelizacija podataka

Iz navedenih zapisa ubačenih u bazu podataka (iz zadatka 2), potrebno je vizuelizovati sledeće podatke:

a) 10 najzastupljenijih mesta koja imaju najveći broj aranžmana u ponudi.
b) Broj aranžmana po mestima.
c) Broj (i procentualni odnos) aranžmana po hotelima sa različitim brojem zvezdica.
d) Broj (i procentualni odnos) svih aranžmana, koje po ceni pripadaju jednom od sledećih opsega:
    - manje ili jednako od 500 eura,
    - između 501 i 1500 eura,
    - između 1501 i 3000 eura,
    - 3000 eura ili više.
e) Broj (i procentualni odnos) aranžmana po tipu usluge koji nude.

Kao rezultat zadatka 3 treba priložiti bazu podataka (iz zadatka 2), realizovane upite i generisane rezultate u vidu grafikona (*charts*). Za grafikone možete koristiti bilo koji alat / biblioteku, a izvoz uraditi kao slike.

Zadatak 4: Implementacija regresije

Realizovati malu aplikaciju koja na osnovu zapisa iz Vaše filtrirane baze podataka primenjuje višestruku linearnu regresiju na nekoliko nezavisnih ulaznih promenljivih i pravi što bolji model zavisnosti između prediktora i ciljne (izlazne) promenljive. Podatke podeliti na skup za treniranje i skup za testiranje, a obučavanje realizovati korišćenjem gradijentnog spusta. Ulazni atributi (*features*) koje možete analizirati mogu biti: mesto, broj noćenja, broj zvezdica hotela i tip usluge.

Ciljna promenljiva treba da bude cena aranžmana. Aplikacija treba da na osnovu ulaznih promeljivih koje korisnik treba da unese preko forme i realizovanog modela, prikaže prediktivnu vrednost aranžmana.

U ovom zadatku nije dozvoljeno korišćenje gotovih funkcija iz neke biblioteke programskog jezika, osim u cilju provere ispravnosti sopstvenih rezultata. Sve funkcije treba da budu samostalno napisane.

Zadatak 5: Implementacija klasifikacije

U okviru iste aplikacije, primeniti još i algoritam K najbližih suseda na osnovu istih ili sličnih ulaznih promenljivih (atributa aranžmana) i na osnovu potpuno iste (filtrirane) baze podataka, kao u zadatku 4. Opseg izlazne vrednosti (cene aranžmana) podeliti na nekoliko klasa, kao što je na primer navedeno u zadatku 3.d).

Kao rezultat zadataka 4 i 5 treba priložiti programski kod realizovane aplikacije ili aplikacija (implementacije realizovanih finalnih modela, procedure za obučavanje, sve realizovane, i eventualno pomoćne funkcije i klase, koje su korišćene). Takođe, priložiti izveštaje sa kratkim komentarom o realizovanim implementacijama, šta ste sve probali da biste došli do finalne implementacije i koji su sve dobijeni rezultati kojima ste evaluirali modele. U ova dva zadatka takođe je poželjno koristiti vizuelizaciju podataka i grafikone (samo potpuno urađeni zadaci donose najveći broj poena!).