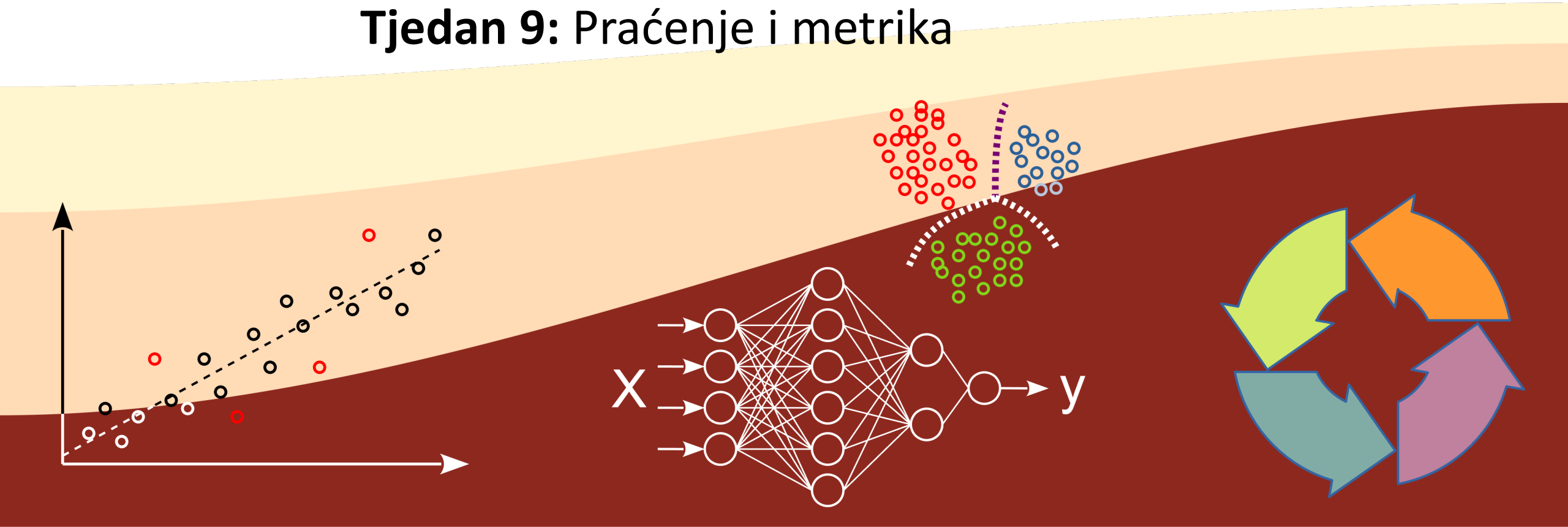
Arhitektura i Razvoj Inteligentnih Sustava

Tjedan 9: Praćenje i metrika



Creative Commons

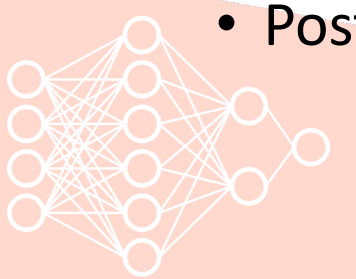
- slobodno smijete:
  - dijeliti — umnožavati, distribuirati i javnosti priopćavati djelo
  - prerađivati djelo
- pod sljedećim uvjetima:
  - imenovanje: morate priznati i označiti autorstvo djela na način kako je specificirao autor ili davatelj licence (ali ne način koji bi sugerirao da Vi ili Vaše korištenje njegova djela imate njegovu izravnu podršku).
  - nekomercijalno: ovo djelo ne smijete koristiti u komercijalne svrhe.
  - dijeli pod istim uvjetima: ako ovo djelo izmijenite, preoblikujete ili stvarate koristeći ga, preradu možete distribuirati samo pod licencom koja je ista ili slična ovoj.

Osvrt na dosadašnje teme

- *Explaineri* i *drift* detektori su bitne komponente
  - Ukoliko su ovakvi modeli u sekvenci sa osnovnim modelom (*inference*), tada se izlaz vidi u odgovoru s kojim se pozvao cjevovod na Seldon Core v2 serveru
  - Ukoliko su ovakvi modeli u paraleli sa osnovnim modelom, tada se odgovori mogu potražiti u Kafka toku podataka – ili korištenjem CLI-a (*pipeline output inspect*)
  - Rezultati ovih modela su informativni – omogućavaju nam reakciju i zahvat na sam model
    - Podešavanje prostora značajki
    - Ponovno učenje modela s novim podacima

Standardna metrika

- ML serveri izlažu REST *endpoint-ove* za metriku
  - OpenMetrics projekt za uspostavu standarda
  - Iniciran od Prometheusa, kako bi se uspostavio jedinstven način dijeljenja metrike
  - Tipovi metrika – *Gauge*, *Counter*, *Histogram*, *Summary*, ...
- Seldon MLServer – posebni *endpoint* za metriku
  - Može se konfigurirati u *settings.json* opisniku
  - Daje osnovnu metriku za modele
    - Broj poziva, trajanje poziva i slično
  - Koristi zajednički *store* za višedretveni način rada
  - Na taj se *endpoint* može spojiti Prometheus i čitati vrijednosti
  - Postoji i *prometheus_client* python modul kojim se to može raditi ručno



Prilagođena (*custom*) metrika

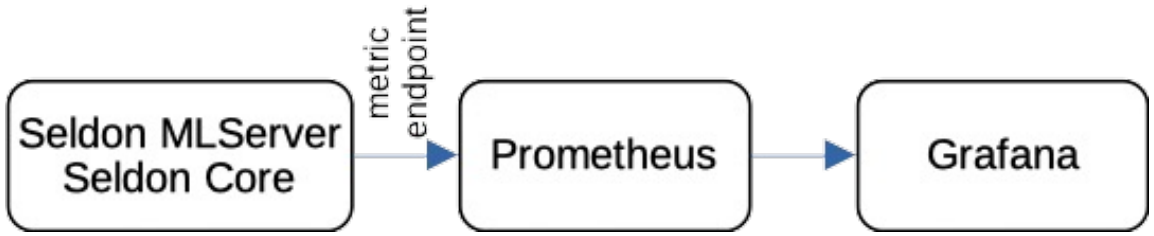
- Seldon ML server omogućava i prilagođenu metriku za naše modele
  - Prilagođeni modeli imaju mogućnost registrirati svoje metrike

    `mlserver.register` `mlserver.log`
  - Ovako definirane metrike će biti dostupne kroz *endpoint* za metriku u obliku histograma
  - Uobičajeno je da se kod učitavanja modela (*load* metoda u prilagođenom modelu) registriraju i prilagođene metrike
  - Kod *predict* metode se zatim logiraju podaci te metrike

- Seldon Core v2
  - Dodatna zajednička standardna metrika za sve modele
  - Brojači (*counter*) i mjerači (*gauge*) – brojanje poziva, vremena odaziva i slično
  - *Hodometer* – komponenta koja skuplja stanja resursa na kubernetes *clusteru* na kojem je Seldon Core instaliran
    - Ta se metrika resursa zatim izlaže kroz *endpoint* za metriku

Kubernetes metrika

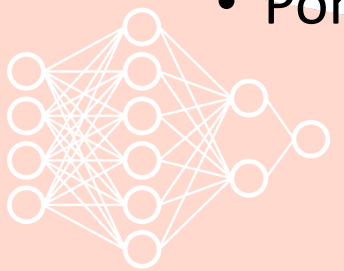
- Generalno
  - Ova se metrika odnosi na performanse modela
  - To nam je bitno da ocijenimo koliko je opterećenje na infrastrukturu, koliko se dugo izvršavaju naši modeli, koliko poziva, ...



Zatvoreni krug zahtjeva

- Korisničke zahtjeve dijelimo na
  - Funkcionalne – što model treba raditi?
    - Funkcionalna metrika se dobiva iz explainera, drift detektora i prilagođene metrike
  - Nefunkcionalne – kako to taj model treba raditi?
    - Brzina, skalabilnost, broj korisnika, broj poziva u jedinici vremena
    - Nefunkcionalna metrika se dobiva iz metrike modela

- Funkcionalna nesukladnost se rješava
  - Intervencijom u podatke i postupak obrade podataka
  - Promjenom pristupa – drugi algoritam, drugi model, isto sučelje (?)
  - Ponovnim učenjem na novijem skupu podataka



Zatvoreni krug zahtjeva

- Nefunkcionalna nesukladnost se rješava
  - Promjenom pristupa – drugi algoritam, drugi model – pokušaj simplifikacije modela drugim algoritmom
  - Redukcija prostora značajki – temeljem rezultata rada explainera
  - Intervencijama u infrastrukturu
    - Korištenje skalabilne i elastične infrastrukture
    - Dodavanje čvorova, dodavanje replika, ...

Ostali podaci

- Recimo da postoji servisni cjevovod iz kojeg zovemo cjevovod u Seldon Core-u
  - U servisnom cjevovodu agregiramo podatke koje zatim upisujemo u razne baze podataka
  - Sve ono što je ostalo – a toga ima mnogo – možemo uhvatiti i obraditi prilagođenim obradama u servisnom cjevovodu
  - Primjer, uobičajen je poziv za pokretanje cjevovoda za učenje u slučaju kada drift detektor utvrdi značajno odstupanje od originalnih klasifikacija u trenutku učenja modela
  - Ovo je podložno određenoj statističkoj relaksaciji – potrebna je određena količina drift detekcija kako bi se ustvrdilo da model nije odgovarajuć
    - U jedinici vremena?

Utjecaj rada inteligentnog servisa na podatke

- Inteligentni servisi svojim radom utječu na svoju okolinu
  - Tim utjecajem, to jest klasifikacijama, predikcijama i odlukama utječu na druge informacijske sustave u koje su uključeni
  - Ali utječu i na same korisnike
    - Forsirano – inteligentni sustav donosi odluku i upravlja
    - Kroz preporuke – inteligentni sustav daje preporuku, a korisnik u konačnici odlučuje
    - Procesno – inteligentno upravljanje usmjerava korisnike i pojačava neke tokove u procesima
    - Podatkovno – recimo kupci se usmjeravaju na određene proizvode – što oni mogu prihvatiti ili ne – ali to može imati utjecaja na kulturu kupca
  - Korisnici u manjoj ili većoj mjeri preuzimaju navike u skladu s interakcijom s inteligentnim sustavom
    - Po onome „brže, lakše, bolje”
    - Po onome „ako je trend, mora da je dobro”

Utjecaj rada inteligentnog servisa na podatke

- No, to u neku ruku mijenja podatke koji su rezultata rada inteligentnog sustava
  - Ako optimiramo proces (pa čak i kroz preporuku), neki tokovi u procesu će postati *sparse*
  - Ako nudimo samo *trendy* robu, sasvim je moguće da će to prouzročiti kanibalizaciju suplementarnih proizvoda
- U konačnici postoji sprega između modela i krajnjeg korisnika
  - Model utječe na korisnika, što se reflektira na podacima – korisnik se približava modelu
  - Korisnik utječe na model, što se reflektira opet na podacima – model se ponovnim učenjem približava korisniku