Sadržaj

Core.clj datoteka je tokom razvoja projekta služila kao početna tačka za pokretanje aplikacije. U početku razvoja, u njoj su implementirane pojedine funkcije koje su učitavale IMDbMovies CSV fajl iz resources, pretvarale prvi reda fajla u header... Kako se projekat razvijao, suština core.clj bila je pisanje komentara, objedinjavanje svih drugih drugih modula u jednu celinu i prikaz razvoja rada na modelu. Međutim, funkcije sa početka su i dalje ostale, pa su zbog organizacije projekta prebačene u cleaning.clj sa svim ostalim funkcijama za čišćenje podataka. U trenutnoj verziji core.clj služi isključivo kao dnevnik razvoja modela - kroz komentare.

U nastavku je objašnjeno šta radi koji modul (namespace) projekta.

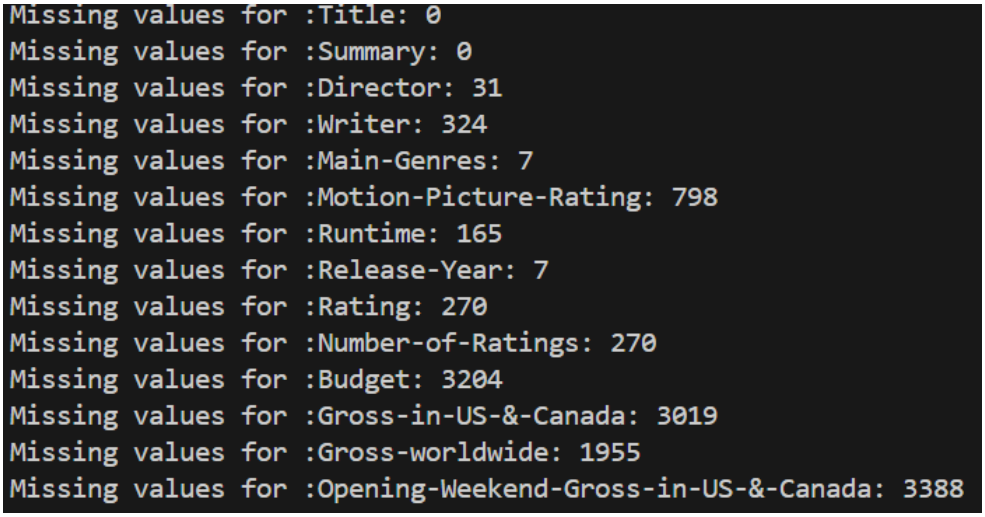
Cleaning.clj

Prvo je implementirana funkcija **load-csv** koja učitava “resources/IMDbMovies.csv”.

process-data:

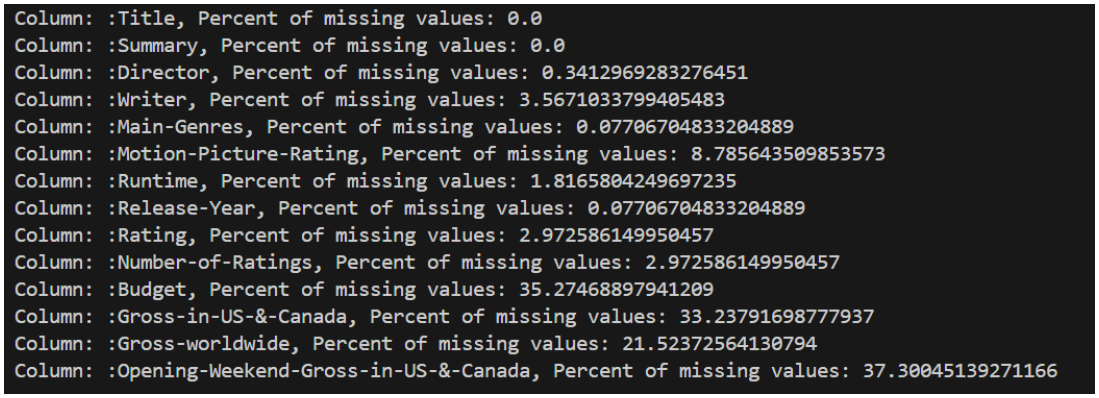
- prvi red iz IMDbMovies (nazive kolona) tumači kao zaglavlje; konvertuje nazive kolona u Clojure keyword-e (convert-to-keywords) i zamenjuje razmake crticama (replace-spaces-with-dashes), npr. "Main Genres" prelazi u :Main-Genres.
- svaki naredni red mapira u mapu preko row-to-map.
- vraća strukturu {:header … :rows …} pogodnu za dalju obradu.

Zatim, za dalju analizu koje kolone zadržati, provereno je koliko nedostajućih vrednosti se nalazi u svakoj koloni.



Slika 1 - Broj NA vrednosti po koloni

A procentualno u odnosu na celu kolonu



Slika 2 - Procenat NA vrednosti po koloni

Kroz komentare se vidi da je odlučeno da se izostave:

- Opening-Weekend-Gross-in-US-&-Canada sa najvećim brojem NA vrednosti;
- Summary, Director, Title i Writer jer su tekstualne varijable sa puno različitih vrednosti pa ih je teško pouzdano transformisati u kategoričke/numeričke;
- Motion-Picture-Rating koji ima 29 različitih vrednosti pa je teško kvalitetno ih prebaciti u kategoričke/numeričke.

Cleaning takođe ima i “pomoćne” funkcije koje su pozvane samo jednom, ali su ostavljene u kodu. Npr. dijagnostikovanje koje sve valute postoje u novčanim kolonama (get-all-currency-prefixes), uzimanje svih različitih vrednosti iz Motion-Picture-Rating (get-all-distinct-rated).

Svrha Cleaning.clj modula je pretvaranje sirovih string kolona u čiste numeričke vrednosti i binarne indikatore, uklanjanje originalnih kolona koje više nisu potrebne i pripremanje doslednog zaglavlja za novi CSV.

Ključne transformacije:

1. **Valute (Budget, Gross-in-US-&-Canada, Gross-worldwide)**
   - detect-currency: prepoznaje valutu po simbolima/skraćenicama;
   - conversion-rates: kursevi prema USD (aproksimativni, fiksni u kodu);
   - parse-budget: izvlači numerički deo iz stringa i konvertuje u USD;
   - clean-budget: dodaje kolone sa sufiksom -Cleaned (npr. :Budget-Cleaned).

Ovo se radi da bi vrednosti u različitim valutama bile uporedive svođenjem na istu valutu (USD koja je bila najčešća).

2. **Ocena (Rating)**
   - parse-rating: iz "4.5/10" uzima 4.5 kao double.
   - clean-rating: dodaje :Rating-Cleaned.

3. **Trajanje (Runtime)**
   - parse-runtime: npr. "2h 12m" u minute (int).
   - clean-runtime: dodaje :Runtime-Cleaned.

4. **Broj glasova (Number-of-Ratings)**
   - parse-num-of-ratings: "34K" -> 34000, "1.2M" -> 1200000.
   - clean-num-of-ratings: dodaje :Num-of-Ratings-Cleaned.

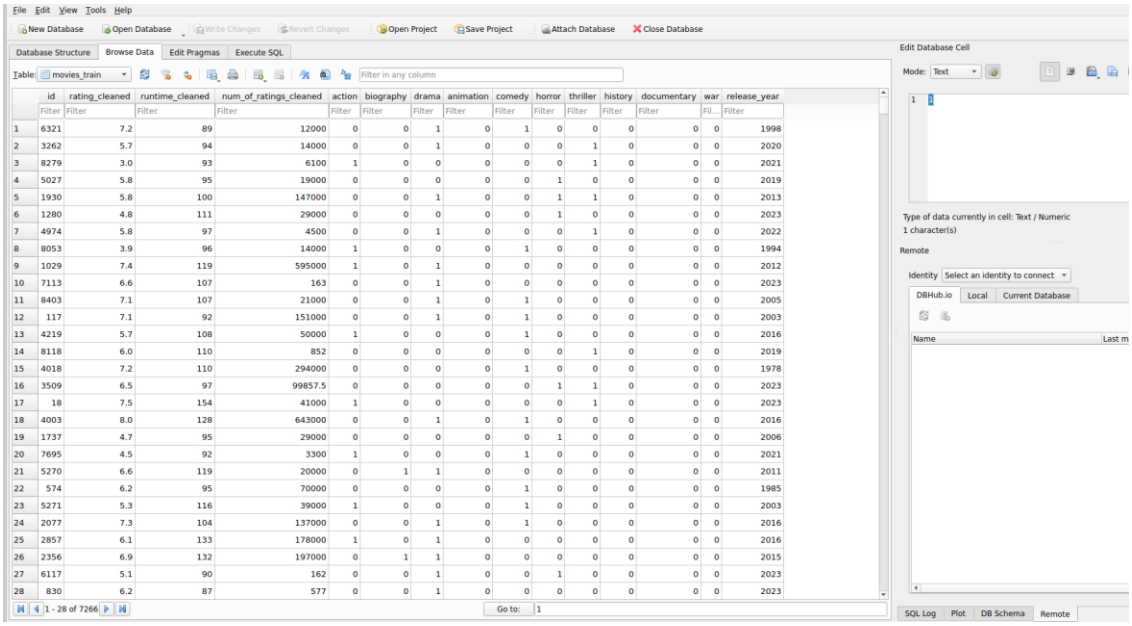
5. **Žanrovi (Main-Genres)**
   - parse-genres: deli po zarezima, uklanja praznine.
   - extract-distinct-genres: jednom izvuče sve žanrove (npr. Action, Drama…).
   - encode-genres / add-genre-columns: "one-hot" kolone 0/1 za svaki žanr.

Za svaki film, prisutni žanrovi dobijaju vrednost 1, a ostali 0. Time se višestruke pripadnosti žanrovima prirodno predstavljaju u obliku više aktivnih kolona u istom redu.

Imputation.clj

U Imputation.clj se učitava CSV koji je prethodno očišćen u cleaning.clj, a zatim se nedostajuće vrednosti (NA) u ciljanim kolonama "amputiraju" srednjom vrednošću po koloni. Nakon amputacije (popunjavanja), dataset se upisuje u novi CSV (finalClean), uz očuvanje istog redosleda kolona.

dbWork.clj



	id	rating_cleaned	runtime_cleaned	num_of_ratings_cleaned	action	biography	drama	animation	comedy	horror	thriller	history	documentary	war	release_year
1	6321	7.2	89	12000	0	0	1	0	1	0	0	0	0	0	1998
2	3262	5.7	94	14000	0	0	1	0	0	0	1	0	0	0	2020
3	8279	3.0	93	6100	1	0	0	0	0	0	1	0	0	0	2021
4	5027	5.8	95	19000	0	0	0	0	0	1	0	0	0	0	2019
5	1930	5.8	100	147000	0	0	1	0	0	1	1	0	0	0	2013
6	1280	4.8	111	29000	0	0	0	0	0	1	0	0	0	0	2023
7	4974	5.8	97	4500	0	0	1	0	0	0	1	0	0	0	2022
8	8053	3.9	96	14000	1	0	0	0	1	0	0	0	0	0	1994
9	1029	7.4	119	595000	1	0	1	0	0	0	0	0	0	0	2012
10	7113	6.6	107	163	0	0	1	0	0	0	0	0	0	0	2023
11	8403	7.1	107	21000	0	0	1	0	1	0	0	0	0	0	2005
12	117	7.1	92	151000	0	0	1	0	1	0	0	0	0	0	2003
13	4219	5.7	108	50000	1	0	0	0	1	0	0	0	0	0	2016
14	8118	6.0	110	852	0	0	0	0	0	0	1	0	0	0	2019
15	4018	7.2	110	294000	0	0	0	0	1	0	0	0	0	0	1978
16	3509	6.5	97	99857.5	0	0	0	0	0	1	1	0	0	0	2023
17	18	7.5	154	41000	1	0	0	0	0	0	1	0	0	0	2023
18	4003	8.0	128	643000	0	0	1	0	1	0	0	0	0	0	2016
19	1737	4.7	95	29000	0	0	0	0	0	1	0	0	0	0	2006
20	7695	4.5	92	3300	1	0	0	0	1	0	0	0	0	0	2021
21	5270	6.6	119	20000	0	1	1	0	0	0	0	0	0	0	2011
22	574	6.2	95	70000	0	0	0	0	1	0	0	0	0	0	1985
23	5271	5.3	116	39000	1	0	0	0	1	0	0	0	0	0	2003
24	2077	7.3	104	137000	0	0	1	0	1	0	0	0	0	0	2016
25	2857	6.1	133	178000	1	0	1	0	0	0	0	0	0	0	2016
26	2356	6.9	132	197000	0	1	1	0	0	0	0	0	0	0	2015
27	6117	5.1	90	162	0	0	1	0	0	1	0	0	0	0	2023
28	830	6.2	87	577	0	0	1	0	0	0	0	0	0	0	2023

Slika 3 - SQLite baza prikaz

Ovaj modul je odradio kompletan rad sa **SQLite bazom**: definisan je db-spec, tabela **movies** je kreirana jednom prilikom inicijalnog set-upa (poziv je ostavljen u kodu), finalni CSV je uvezen u bazu, a obezbeđene su i operacije za pregled i brisanje zapisa. Iz postojeće tabele movies su izvedene **movies_train** i **movies_test** sa istom strukturom (preko PRAGMA table_info), zatim su podaci deterministički pomešani (seed) i podeljeni prema zadatom odnosu, pa upisani pomoću insert-multi!. Rezultati upita se vraćaju kao lower-case mape (rs/as-unqualified-lower-maps) radi jednostavnije obrade u ostatku koda.

Correlation.clj

Korelacije su računate pomoću correlation.clj fajla.
Glavna funkcija u correlation je **correlations-to-target** koja računa Pearsonovu korelaciju između target kolone (rating_cleaned) i svake druge kolone u datasetu (budget_cleaned, release_year…) .

Rezultat je mapa koja kaže koliko je jaka linearna veza svake kolone sa targetom i u kom smeru je ta veza (pozitivna ili negativna).

Ove korelacije su nam potrebne zato što linearni regresioni model pokušava da objasni target (Y) preko linearne kombinacije prediktora (X-ova):

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon$$

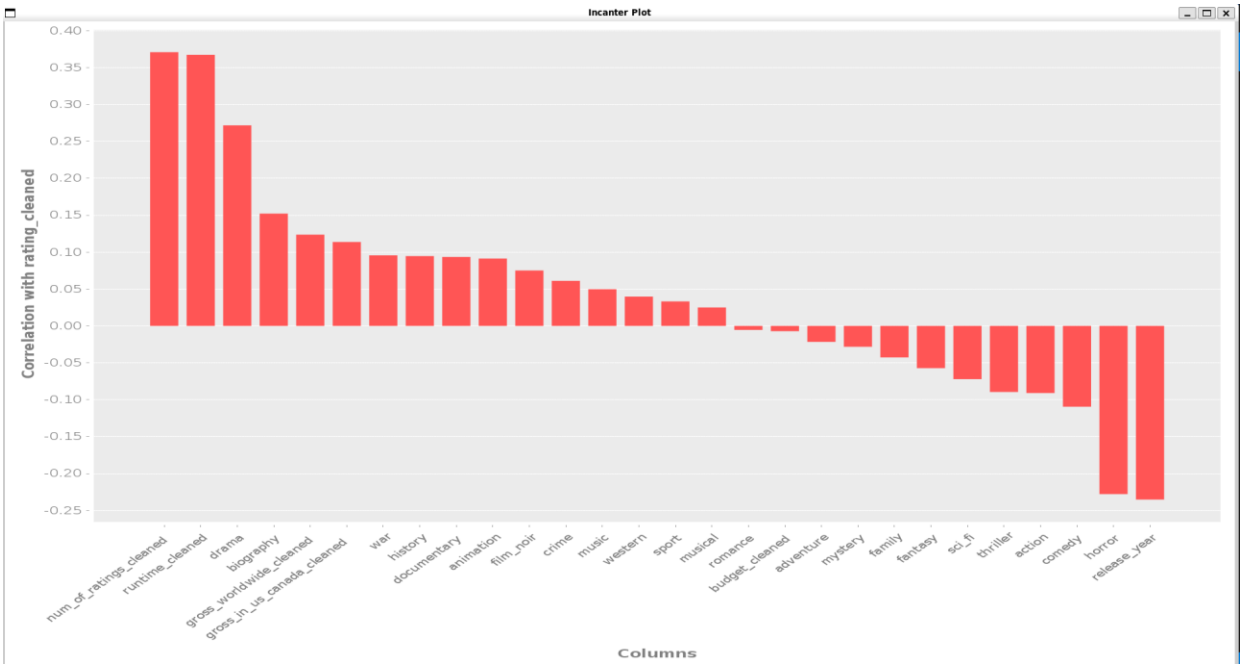
Slika 4 - Linearna regresija formula

Da bi neki prediktor (kolona) bio **koristan u modelu**, on treba da ima **određenu povezanost sa targetom**.

- Ako je korelacija blizu nuli znači da između X i Y nema linearne veze i ta kolona je verovatno **neinformativna** za linearnu regresiju.
- Ako je korelacija visoka (pozitivna ili negativna) ta kolona nosi **jak signal** koji može da pomogne u objašnjenju targeta.

Nakon ispisa korelacije prediktora i izlazne varijable pomoću funkcija print-correlations i analyze-correlation, dobijeno je sledeće:

Slika 5 - Prikaz korelacija svih prektora i zavisne varijable



Slika 6 - Prikaz korelacija svih prektora i zavisne varijable na plot-u

Na osnovu dobijenih korelacija, zadržala sam atribute sa korelacijom većom od |0.08|, osim gross_worldwide_cleaned i gross_in_us_canada_cleaned. Ove dve varijable su isključene jer su od početka imale najveći broj nedostajućih vrednosti (NA), što bi moglo da naruši konzistentnost i pouzdanost modela.

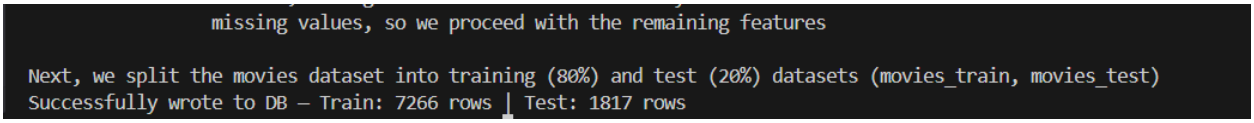
Nakon ovoga sam definisala funkciju koja proverava **multikolinearnost nezavisnih varijabli**. Multikolinearnost nastaje kada su dva ili više nezavisnih (prediktorskih) varijabli međusobno snažno korelisane. U regresiji želimo da svaka varijabla „donosi“ nezavisnu informaciju, a kada se prediktori kreću zajedno, model teško razdvaja njihov pojedinačni efekat.

Kako bih ispitala **multikolinearnost**, računala sam korelacije između svih zadržanih prediktora (bez ciljne i ID kolone) i izdvajala parove sa $|r| \geq 0.8$. U projektu to radim funkcijom **multicollinear-pairs** sa pragom 0.8.

Jedini par sa $|r| > 0.8$ bio je gross_worldwide_cleaned <> gross_in_us_canada_cleaned ($r \approx 0.9165$).
Pošto su sve gross* promenljive ionako isključene ranije zbog visokog procenta NA vrednosti, u zadržanom skupu prediktora ne postoje parovi koji premašuju prag $|r| \geq 0.8$.
Zaključak: rizik od štetne multikolinearnosti u odabranim karakteristikama **je nizak**, pa se može nastaviti sa modeliranjem bez dodatnih intervencija po tom osnovu.

Napomena: u kasnijim iteracijama, nakon benchmarking-a funckija multicollinear-pairs je izmenjena tako da radi samo sa varijablama koje su nakon računanja korelacija ušle u izbor. To znači da nakon njenog izvršavanja, ne vraća **ni jedan par čija je kolinearnost veća od 0.8**. Izmena ove funkcije je rađena iz razloga što se predugo izvršavala.

Nakon odabira atributa koji će se koristiti u modelu, delimo dataset Movies na train i test datasetove. Model se uči na *train* skupu, dok se *test* koristi isključivo za predikciju i završnu evaluaciju na podacima koje model ranije „nije video“.
Kako bismo to postigli, koristimo prvo **create-empty-like** funkciju kako bismo kreirali prazne tabele movies_train i movies_test sa istim tipovima kolona kao movies (samo za selected-cols, preko PRAGMA table_info).
Nakon toga, koristimo fn **split-train-test-rows** kojoj prosleđujemo redove iz Movies tabele u bazi, ratio (0.8) i seed (nasumično 26). Fn vraća {:train [...] :test [...]} uz očuvanje nasumičnog, ali determinističkog redosleda.
Zatim se vrednosti konačno ubacuju pozivom insert-multi! u odgovarajuće tabele, pod transakcijom, pozivom fn **insert-data-train-test.**
S obzirom da se ova fn poziva samo jednom kako bi ubacila vrednosti u odgovarajuće tabele, ispis je sledeći:



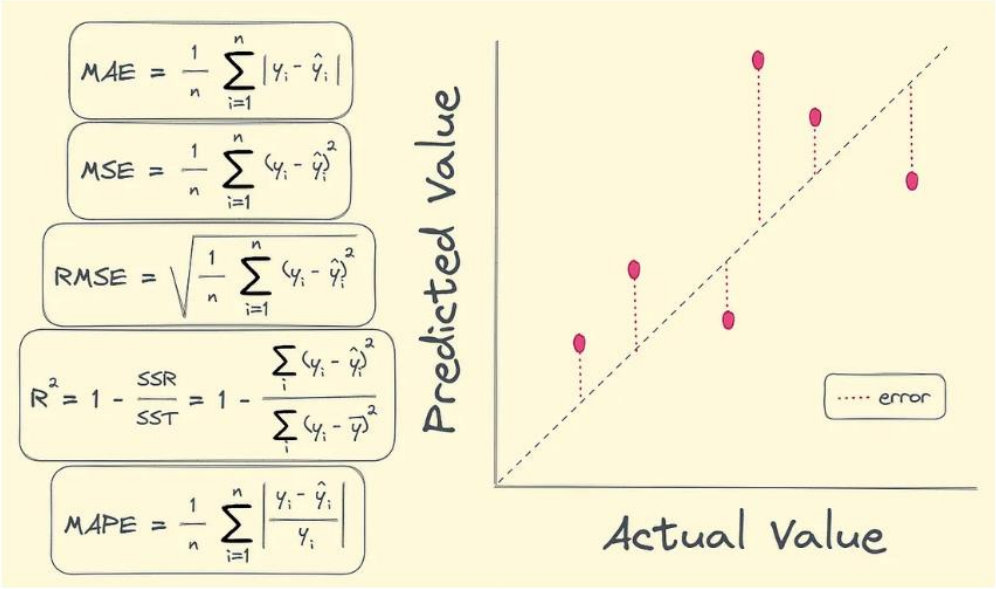
Slika 7 - Prikaz podele Movies tabele na train i test

Lm.clj

Nakon podele dataseta na train i test deo, zvanično je omogućeno da se krene sa kreiranjem modela linearne regresije.

lm.clj pokriva ceo tok: **fit-stats** na trening skupu za svaku numeričku kolonu izračuna μ (mu) i sd. μ (mu) je oznaka za aritmetičku sredinu (prosek), a sd je standardna devijacija uzorka (računata sa n-1). Po potrebi se pre toga primeni log1p (ln(1+x)). **transform-row** na pojedinačan red primeni istu proceduru (opcionalni log1p, pa z-score: (x−μ)/sd). Zatim **add-interactions** dodaje interakciju :num_of_ratings_cleaned_x_release_year kao proizvod već standardizovanih num_of_ratings_cleaned i release_year. **fit-lm** koristi Incanter (stats/linear-model) tako što formira vektor cilja i matricu obeležja i nauči linearnu regresiju (intercept + koeficijenti). **train-model** sve to orkestrira: fituje preprocesiranje, transformiše train redove (uključujući interakciju) i trenira model; vraća stats, sam model i funkciju transform-row za doslednu primenu na testu i u produkciji. **predict-y** računa predikciju kao intercept + Σ(β·x). **evaluate** vraća RMSE, MAE, R^2. **eval-model** pripremi test, odštampa metrike i tabelu značajnosti (β, t, p), sortiranu po p-vrednosti i jasno obeleži kandidate za izbacivanje (p>0.05). Na kraju, **train-and-save** upiše artefakt u resources/lm-artifact.edn (:selected-cols, :stats, :intercept, :betas) da bi server mogao identično da transformiše ulaz i koristi iste koeficijente pri predikciji.

Evalutivne metrike se računaju na sledeći način:



Slika 8 - Prikaz formula za evalutivne metrike

```clojure
(defn evaluate
  "Computes RMSE, MAE, and R² between true and predicted targets"
  [y-test y-predicted]
  (let [n (count y-test)
        ybar (/ (reduce + y-test) n)
        abs-errors (map (fn [a b] (Math/abs (double (- (double a) (double b))))) 
                        y-test y-predicted)
        mae  (/ (reduce + abs-errors) n)
        ssres (reduce + (map (fn [a b] (let [e (- a b)] (* e e))) y-test y-predicted))
        sstot (reduce + (map (fn [a] (let [d (- a ybar)] (* d d))) y-test))]
    {:rmse (Math/sqrt (/ ssres n))
     :mae mae
     :r2   (if (zero? sstot)
             0.0
             (- 1.0 (/ ssres sstot)))}))
```

Slika 9 - Prikaz implementacije evalutivnih metrika

Istorija razvoja ovog modela je sledeća:

Nakon testiranja nad „izmišljenim“ podacima, model je treniran nad pravim train podacima iz baze. Metrike evaluacije za tako napravljen model su sledeće:

```
Metrics: {:rmse 0.8383800663333316, :mae 0.6119669954277123, :r2 0.3629152548181541}
Betas of model:
Intercept: 30.42253032044391
num_of_ratings_cleaned 0.000002
runtime_cleaned        0.011005
drama                  0.418107
biography              0.263628
war                    0.157485
history                0.007088
documentary            1.343869
animation              0.731066
thriller               0.017953
action                 -0.206146
comedy                 -0.049306
horror                 -0.299066
release_year           -0.012734
```

Slika 10 - Početne metrike

RMSE i MAE su mere veličine greške: **MAE** je prosečno apsolutno odstupanje između predikcije i stvarne vrednosti (svaka greška se računa “koliko je daleko”, bez znaka), dok **RMSE** računa isto to ali jače kažnjava veće promašaje (kvadrira greške pre proseka pa vadi koren). Manji RMSE/MAE su bolji. **R^2** govori koliki deo “šarenila” (varijacije) u ciljnoj vrednosti model uspeva da objasni u odnosu na trivijalnu strategiju “uvek predviđaj prosek”; veći R^2 je bolji (1 je savršeno, 0 je kao pogađanje proseka, negativno je lošije od proseka).

U ovom modelu, **MAE ≈ 0,612** znači da je tipična greška oko 0,61 ocene: ako je stvarna ocena 7, često će predikcija biti oko 6,4-7,6. **RMSE ≈ 0,838** je veći od MAE, što sugeriše da povremeno postoje i malo veći promašaji (npr. oko 1 ocene ili više), pa je "kvadratna" prosečna greška osetljivija. **R^2 ≈ 0,3629** znači da model smanjuje ukupnu kvadratnu grešku za oko 36% u odnosu na naivno predviđanje konstantnog proseka svih ocena; deo signala (žanrovi, trajanje, godina…) je uhvaćen, ali značajan deo varijacije ostaje neobjašnjen.

U ovom skupu podataka postoje dve vrste ulaza: binarne i kontinualne varijable. **Binarne (0/1)** označavaju da li film pripada nekom žanru poput drama, horror, comedy… i koeficijent (β) ovih varijabli se čita kao razlika u prosečnoj oceni između filmova koji imaju taj žanr (1) i onih koji ga nemaju (0), uz sve ostalo isto. **Kontinualne varijable** (npr. runtime u minutima, release_year, num_of_ratings_cleaned) mogu da uzmu mnogo vrednosti i njihov β kaže kolika je promena očekivane ocene kada ta varijabla poraste za 1 (npr. +1 minut trajanja).

Npr: za film iz 2010. sa trajanjem 120 min koji je sa drama (drama=1, ostali žanrovi 0), predikcija po ovako dobijenim koeficijentima je približno 6,57: 30,42 (intercept) + 0,0110×120 - 0,012734×2010 + 0,418 (drama) ≈ 6,57. Ako se isti film označi kao horror i comedy (horror=1, comedy=1, drama=0), onda se ocena spusti za oko 0,299 (horror) i još oko 0,049 (comedy), pa predikcija bude oko 6.28. Ovo lepo ilustruje da se efekti binarnih žanrova sabiraju, a da kontinualne varijable daju male, ali kumulativne doprinose po jedinici (minuti, godine).

Intercept (~30) nije „ocena 30", već vrednost koju bi model dao kada su sve varijable jednake nuli (godina=0, trajanje=0…), što je tačka van realnog domena. Zato je potrebno kontinualne varijable standardizovati, čime nula postaje „prosečna vrednost", pa intercept postaje približno „prosečna ocena tipičnog filma", a koeficijenti kontinualnih varijabli postaju lakše uporedivi.

Sada je cilj **poboljšati model**, jer trenutna verzija koristi sirove vrednosti numeričkih varijabli koje imaju različite raspone i izraženu desnu nagnutost u distribuciji. **Desna nagnutost** znači da najveći broj posmatranja ima male vrednosti, dok mali broj primera zauzima ekstremno velike vrednosti, pa rep distribucije ide udesno. Konkretno, broj ocena filmova ima vrlo veliki opseg i snažnu desnu nagnutost, što dovodi do toga da linearni model teško prepoznaje obrasce.

Kako bih to ispravila, planiram da primenim log-transformaciju nad ovom kolonom. Log-transformacija podrazumeva primenu logaritamske funkcije na vrednosti, čime se velike vrednosti kompresuju, a male vrednosti zadržavaju veću diferencijaciju. Na taj način ublažava se uticaj ekstremnih vrednosti i postiže se ravnomernija distribucija podataka, što olakšava modelu da uoči pravilnosti. Pored toga, uvedena će biti i standardizacija numeričkih varijabli poput trajanja filma i godine izlaska. Standardizacija ih svodi na istu skalu (prosek nula, standardna devijacija jedan), čime se sprečava da varijable sa većim opsegom imaju nesrazmerno veliki uticaj na model. Kombinacija log-transformacije i standardizacije trebalo bi da obezbedi stabilniji model i poboljša njegove performanse.

Nakon standardizovanja numeričkih varijabli metrike izgledaju ovako:

```
Metrics: {:rmse 0.84237502127644, :mae 0.6032128852246047, :r2 0.35682925915446173}
Betas of model:
Intercept: 6.38789105687897
num_of_ratings_cleaned 0.331639
runtime_cleaned        0.249676
drama                  0.360590
biography              0.179491
war                    0.155893
history                -0.002014
documentary            1.565325
animation              0.651196
thriller               -0.029557
action                 -0.278332
comedy                 -0.143183
horror                 -0.382415
release_year           -0.198765
```

Slika 11 - Metrike nakon standardizovanja numeričkih variajbli

Performanse su skoro iste. RMSE je blago porastao (0.838→**0.842**), MAE se malo popravio (0.612→**0.603**), a R^2 je blago pao (0.363→**0.357**).

Najveća promena je u koeficijentima zbog skaliranja/log-transformacije numeričkih kolona: intercept je sa ~30.4 pao na ~**6.39**. Ovo je povećalo interpretabilnost. Kad su numeričke kolone centrirane, 0 znači "prosečna vrednost". Zato je intercept ≈ očekivana ocena filma kad su numerike na proseku, a žanrovi = 0 (nema tog žanra), što je smisleno.
β za numeričke feature-e sada znače promenu ocene po +1 SD te kolone. Znakovi su uglavnom ostali slični: dokumentarci i animacija i dalje pozitivni, action i horror negativni, thriller i history praktično nula. Release_year je postao izraženije negativan (sad na standardizovanoj skali), a uticaj num_of_ratings i runtime je porastao jer su sada na uporedivoj skali (do sada su vrednosti ovih feature-a bili mnogo veći od ostalih).

Zaključak: interpretacija je jasnija, ali ukupni kvalitet modela je ostao približno isti.

Sledeći korak podrazumeva analizu **p-vrednosti** kako bi se utvrdilo da li su sve uključene varijable zaista neophodne u modelu.

```
Metrics: {:rmse 0.84237502127644, :mae 0.6032128852246047, :r2 0.35682925915446173}
Intercept: 6.38789105687897

Significance (sorted by p desc):
history                b=-0.002014  t=-0.036  p=0.9715   <-- kandidat za brisanje
thriller               b=-0.029557  t=-1.068  p=0.2856   <-- kandidat za brisanje
war                    b= 0.155893  t= 2.181  p=0.0292
biography              b= 0.179491  t= 4.428  p=0.0000
comedy                 b=-0.143183  t=-5.741  p=0.0000
num_of_ratings_cleaned b= 0.331639  t= 32.484  p=0.0000
runtime_cleaned        b= 0.249676  t= 23.336  p=0.0000
drama                  b= 0.360590  t= 15.016  p=0.0000
documentary            b= 1.565325  t= 16.902  p=0.0000
animation              b= 0.651196  t= 13.977  p=0.0000
action                 b=-0.278332  t=-10.808  p=0.0000
horror                 b=-0.382415  t=-12.476  p=0.0000
release_year           b=-0.198765  t=-20.340  p=0.0000
```

Slika 12 - Analiza p-vrednosti

Do ovih vrednosti je jednostavno doći, jer ih funkcija stats/linear-model automatski vraća kao rezultat. Značenje ovih metrika:

- **b (beta koeficijent)** pokazuje **pravac i jačinu veze** između tvoje varijable i cilja (ovde rating_cleaned).Ako je **pozitivno**: što više te varijable, raste ocena. Ako je **negativno**: što više te varijable, pada ocena.
- **t (t-statistika)** govori koliko je taj koeficijent "jak" u poređenju sa šumom (nasumičnim varijacijama). Što je veći apsolutno, to je pouzdanije da efekat stvarno postoji.
- **p (p-vrednost)** je verovatnoća da se ovakav efekat pojavi **slučajno**. Ako je **p < 0.05**: skoro sigurno efekat postoji (statistički značajan). Ako je **p > 0.05**: nema dokaza da ta varijabla ima stvarnog uticaja (može biti brisana).

S obzirom da je za history p=0.9715, a za thriller=0.2856, verovatnoće da ove varijable nemaju uticaja na **rating** su velike, pa ćemo ih isključiti dalje iz modela.

```
Metrics: {:rmse 0.8421242484149689, :mae 0.6028399476585521, :r2 0.35721214268080137}
Intercept: 6.374684818301902

Significance (sorted by p desc):
war                    b= 0.159032  t= 2.230  p=0.0258
biography              b= 0.184599  t= 4.675  p=0.0000
comedy                 b=-0.133676  t=-5.742  p=0.0000
num_of_ratings_cleaned b= 0.331403  t= 32.530  p=0.0000
runtime_cleaned        b= 0.250123  t= 23.613  p=0.0000
drama                  b= 0.366744  t= 15.754  p=0.0000
documentary            b= 1.576331  t= 17.129  p=0.0000
animation              b= 0.657774  t= 14.245  p=0.0000
action                 b=-0.276176  t=-10.759  p=0.0000
horror                 b=-0.381450  t=-12.452  p=0.0000
release_year           b=-0.199682  t=-20.514  p=0.0000
```

Slika 13 - p-vrednosti nakon isključivanja history i thriller

Nakon što smo isključili history i thriller, model se poboljšao za nijansu:
RMSE: 0.8423 -> **0.8421**
MAE: 0.6032 -> **0.6028**
R^2: 0.3568 -> **0.3572**

Da bi se dodatno poboljšale performanse modela, iskorišćen je koncept ***feature interaction*** - uvođenje nove varijable nastale kao proizvod dve postojeće. Na taj način model ne posmatra efekte svake varijable izolovano, već može da uči da njihov uticaj zavisi jedan od drugog. Na primer, efekat broja ocena na rejting nije isti za film iz 1970. i za film iz 2020, već zavisi od godine izlaska. Dodavanjem ovakvih interakcionih termina prelazi se iz prostog aditivnog modela, koji samo sabira uticaje, u model sa tzv. "cross-terms", koji realističnije hvata zavisnost i kontekst međusobnih efekata varijabli.

Sa dodatom varijablom num_of_ratings_cleaned_x_release_year, metrike modela su sledeće:

```
Metrics: {:rmse 0.8253926905376342, :mae 0.593710890657903, :r2 0.3825005776475592}
Intercept: 6.332923055235496

Significance (sorted by p desc):
war                                 b= 0.115677  t= 1.648  p=0.0993   <-- kandidat za brisanje
comedy                              b=-0.118752  t=-5.184  p=0.0000
biography                           b= 0.202210  t= 5.207  p=0.0000
num_of_ratings_cleaned              b= 0.381827  t= 36.330  p=0.0000
runtime_cleaned                     b= 0.237642  t= 22.753  p=0.0000
drama                               b= 0.386160  t= 16.847  p=0.0000
documentary                         b= 1.515876  t= 16.738  p=0.0000
animation                           b= 0.663098  t= 14.605  p=0.0000
action                              b=-0.250225  t=-9.894  p=0.0000
horror                              b=-0.368796  t=-12.240  p=0.0000
release_year                        b=-0.237863  t=-24.102  p=0.0000
num_of_ratings_cleaned_x_release_year    b=-0.160510  t=-15.843  p=0.0000
```

Slika 14 - Metrike nakon dodavanja interakcije

Dakle, primećujemo da su se performanse modela poboljšale (**RMSE**: 0.8421 -> **0.8253**, **MAE**: 0.6028 -> **0.5937**, **R^2:** 0.3572 -> **0.3825**), ali i da je p-vrednost za varijablu *War* porasla na 0.0993, jer novi feature preuzima deo varijanse koju je ranije objašnjavao *War*, pa samim tim opada i njegova statistička značajnost.

```
Metrics: {:rmse 0.8257507525479505, :mae 0.5942802191058457, :r2 0.38196470897505186}
Intercept: 6.33559058963957

Significance (sorted by p desc):
biography                              b= 0.200134  t= 5.155  p=0.0000
comedy                                 b=-0.120563  t=-5.269  p=0.0000
num_of_ratings_cleaned                 b= 0.381784  t= 36.322  p=0.0000
runtime_cleaned                        b= 0.239112  t= 22.975  p=0.0000
drama                                  b= 0.387143  t= 16.894  p=0.0000
documentary                            b= 1.518322  t= 16.765  p=0.0000
animation                              b= 0.662648  t= 14.594  p=0.0000
action                                 b=-0.250662  t=-9.910  p=0.0000
horror                                 b=-0.370552  t=-12.305  p=0.0000
release_year                           b=-0.239400  t=-24.364  p=0.0000
num_of_ratings_cleaned_x_release_year  b=-0.161161  t=-15.918  p=0.0000
```

Slika 15 - Metrike nakon izbacivanja War

Iako je p-vrednost za *War* bila iznad praga značajnosti, njegovo uklanjanje dovelo je do pogoršanja performansi modela (**RMSE**: 0.8253 -> **0.8257**, **MAE**: 0.5937 -> **0.5942**, **R^2:** 0.3825 -> **0.3819**),. To pokazuje da *War* ipak nosi određenu informaciju korisnu za predikciju i, iako pojedinačno nije statistički značajan, u kombinaciji sa ostalim varijablama doprinosi boljoj ukupnoj tačnosti modela, mada s obzirom da je razlika zanemarljiva odlučila sam da je izostavim iz završnog skupa prediktora radi jednostavnosti i interpretabilnosti modela.

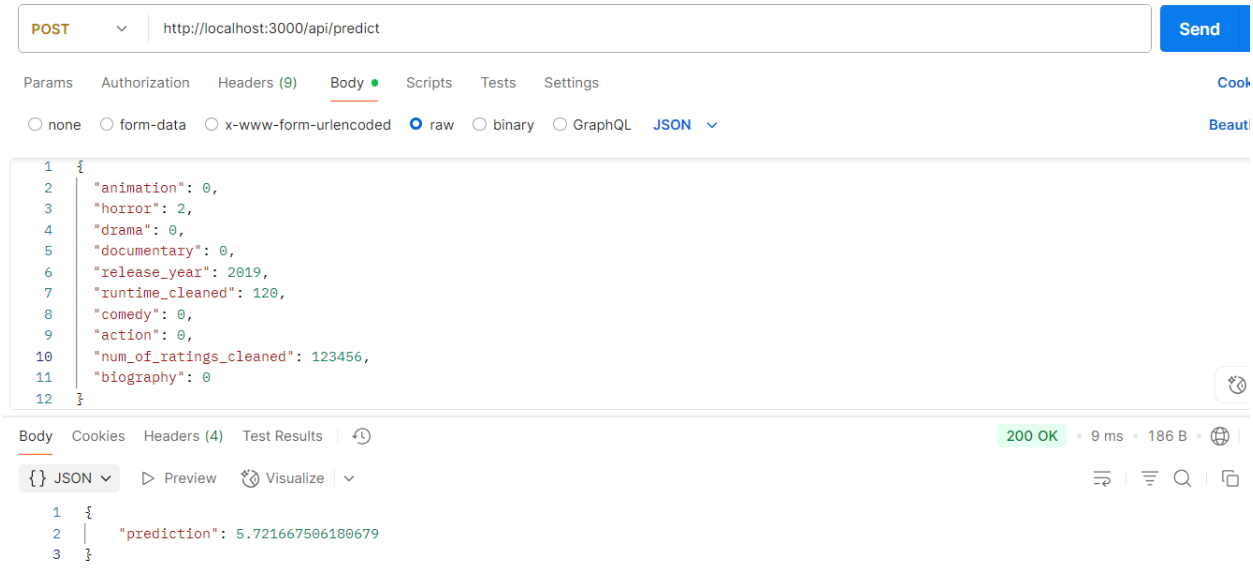
Konačni skup odabranih prediktora obuhvata sledeće varijable: :**num_of_ratings_cleaned, :runtime_cleaned, :drama, :biography, :documentary, :animation, :action, :comedy, :horror, :release_year** i interakcioni termin :**num_of_ratings_cleaned_x_release_year**. To znači da model koristi kombinaciju numeričkih osobina (npr. broj ocena, trajanje i godina izlaska), binarnih indikatora žanrova (poput drama, komedija ili horor), kao i interakciju između broja ocena i godine izlaska, kako bi što preciznije predvideo rejting filma.
Na kraju, formula linearne regresije dobija sledeći oblik:

$$rating = 6.33 + 0.20 \cdot Biography - 0.12 \cdot Comedy + 0.38 \cdot Num\_of\_ratings + 0.24 \cdot Runtime + 0.39 \cdot Drama + 1.52 \cdot Documentary + 0.66 \cdot Animation - 0.25 \cdot Action - 0.37 \cdot Horror - 0.24 \cdot Release\_year - 0.16 \cdot (Num\_of\_ratings \times Release\_year)$$

gde je 6.33 vrednost intercepta. Ovakav model pokazuje kako se rejting filma menja u zavisnosti od pojedinačnih karakteristika i njihovih interakcija, a znak i veličina koeficijenata ukazuju na pravac i jačinu uticaja svake varijable.

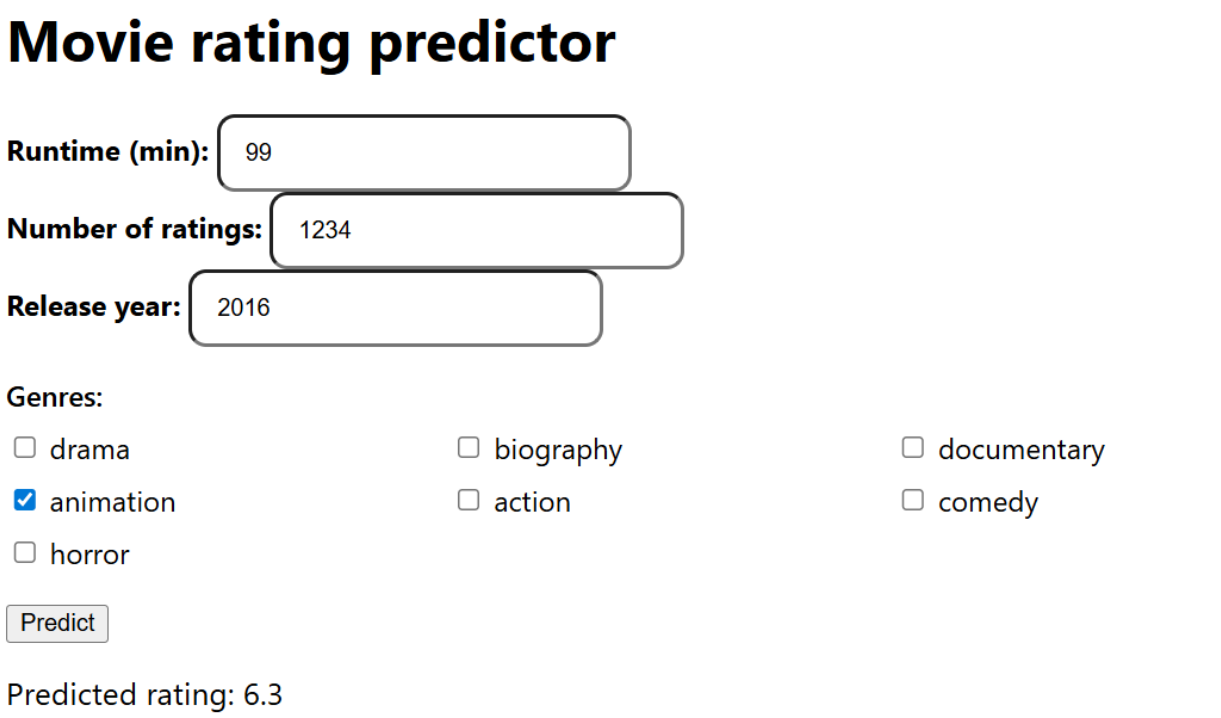
Server testiranje

Testiranje je prvo bilo putem Postmana, a kasnije i preko fronta.



Slika 16 - Prikaz testiranja kroz Postman

Frontend



Slika 17 - Prikaz frontenda

Frontend je minimalističan, rađen u **HTML/CSS + ClojureScript**, i predstavlja jednostavnu formu koja prima **runtime**, **number of ratings** i **release year** (HTML5 validacija: type="number", min/max, pattern="\d+", reportValidity()), uz višestruki izbor **žanrova** preko checkbox-ova (mapirani na 0/1). Na submit, ClojureScript (u app.core) čita vrednosti, sklapa **JSON** payload (runtime_cleaned, num_of_ratings_cleaned, release_year + binarni žanrovi), i šalje ga ka backendu na POST /api/predict preko fetch. Ako odgovor stigne i sadrži rezultat, prikazuje se tekst **"Predicted rating: X.Y"** (zaokruženo na jednu decimalu); ako validacija padne ili mrežni/HTTP poziv ne uspe, poruka o grešci se ispisuje u **#errors**, a rezultat se briše iz **#prediction**. Formu prate male UX sitnice: naglašavanje neispravnih polja (input:invalid), konzolni log ulaza ("Input (JSON)"), i čist, responsivan layout sa gridom za žanrove.

Server.clj

Ovaj namespace podiže HTTP servis i izlaže **POST /api/predict**: pri startu učitava model iz resources/lm-artifact.edn i čuva ga u artifact* (atom). Na zahtev, telo JSON-a se parsira, ulaz se standardizuje preko lm/transform-row i proširi interakcijom lm/add-interactions, potom se predikcija računa kao intercept + Σ(β·x) i ograničava na **[1,10].** Radi dijagnostike, print-summary ispisuje z-vrednosti i doprinose svake osobine (β·x). Uspešan odgovor je JSON oblika {"prediction": <broj>}, a greške se hvataju i vraća se 400 sa porukom. Uključen je CORS (dozvoljeni origin-i npr. http://localhost:8010 i svi preko regex-a), što omogućava da frontend iz odvojenog repoa direktno poziva API. Sve ostale rute vraćaju 404. -main startuje Jetty na portu 3000 i server je spreman za pozive.

Config.clj

Ovaj fajl centralizuje konfiguraciju modela. **all-predictors** je katalog svih kandidata za ulazne kolone (numeričke kolone i one-hot žanrovi) koje sistem može da koristi. **feature-columns** je početni, uži skup korišćen u ranim treninzima/evaluacijama, dok **final-feature-columns** predstavlja zvaničan konačni skup feature-a - uključuje i interakciju :num_of_ratings_cleaned_x_release_year koju tokom preprocesiranja dodaje lm/add-interactions. **target-col** je zavisna promenljiva (:rating_cleaned). Redosled i nazivi moraju biti usklađeni sa šemom u bazi i preprocesiranjem; izmene ovih lista + ponovno treniranje automatski se odražavaju u lm-artifact.edn

Bench.clj

U projektu se koristi Criterium za pouzdano merenje brzine funkcija. Umesto jednokratnog merenja, kod se pokreće više puta i računa se prosečno vreme, pa su rezultati stabilniji i uporedivi. Na osnovu tih merenja unapređene su funkcije **correlations-to-target** i **multicollinear-pairs** (optimizovan način računanja, manje nepotrebnog rada), što je dovelo do primetnog smanjenja vremena izvršavanja. U dokumentaciji se beleže ključne vrednosti pre/posle i kao glavna metrika koristi se prosečno vreme izvršavanja.

Za fju correlations-to-target:

```
(defn correlations-to-target
  "Calculates Pearson correlations between a target column and all other columns in the dataset"
  [data target-col]
  (let [cols (remove #{target-col :movies/id} (keys (first data)))
        target-vec (mapv target-col data)]

         (into {} (map (fn [c] [c (stats/correlation target-vec (mapv c data))]) cols))))
```

Slika 18 - Prikaz početne fje correlations-to-target

```
(defn correlations-to-target
  "Calculates correlations between a target column and each column in cols"
  [data target-col cols]
  (let [target-vec (mapv target-col data)]
    (into {}
          (pmap (fn [c] [c (stats/correlation target-vec (mapv #(get % c) data))]) cols))))
```

Slika 19 - Prikaz ispravljene fje corelations-to-target

Druga verzija radi brže zato što **pmap** računa korelacije za više kolona istovremeno (koristi više jezgara), a **mapv #(get % c)** data odmah pravi gotov vektor brojeva bez dodatnih koraka i usporavanja.

```
;;(crit/bench (corr/correlations-to-target data cfg/target-col cfg/feature-columns))

; Evaluation count : 300 in 60 samples of 5 calls.
;             Execution time mean : 203.053769 ms
;     Execution time std-deviation : 10.287597 ms
;    Execution time lower quantile : 196.741892 ms ( 2.5%)
;    Execution time upper quantile : 225.887661 ms (97.5%)
;                    Overhead used : 12.458469 ns

;;After improvement
(crit/bench (corr/correlations-to-target data cfg/target-col cfg/feature-columns))
; Evaluation count : 1380 in 60 samples of 23 calls.
;             Execution time mean : 51.496042 ms
;     Execution time std-deviation : 5.258600 ms
;    Execution time lower quantile : 45.500580 ms ( 2.5%)
;    Execution time upper quantile : 62.259118 ms (97.5%)
;                    Overhead used : 8.977870 ns
```

Slika 20 - Prikaz poboljšanja za fju correlations-to-target

Fja multicollinear-pairs:

```
(defn multicollinear-pairs
  [data threshold]
  (let [cols (vec (remove #{:rating_cleaned :id} (keys (first data))))
        n    (count cols)]
    (->> (for [i (range n), j (range (inc i) n)]
           (let [xi (mapv (nth cols i) data)
                 xj (mapv (nth cols j) data)
                 r  (stats/correlation xi xj)]
             [(nth cols i) (nth cols j) r]))
         (filter #(>= (Math/abs (nth % 2)) threshold)))))
```

Slika 21 - Prikaz prvobitne fje multicollinear-pairs

```
(defn multicollinear-pairs
  [data threshold cols]
  (let [cols   (vec cols)
        col-v (into {} (map (fn [c] [c (mapv #(double (get % c)) data)]) cols))
        n      (count cols)]
    (->> (for [i (range n), j (range (inc i) n)]
           (let [ci (nth cols i)
                 cj (nth cols j)
                 r  (stats/correlation (col-v ci) (col-v cj))]
             (when (>= (Math/abs r) threshold) [ci cj r])))
         (keep identity)
         vec)))
```

Slika 22 - Prikaz ispravljene fje multicollinear-pairs

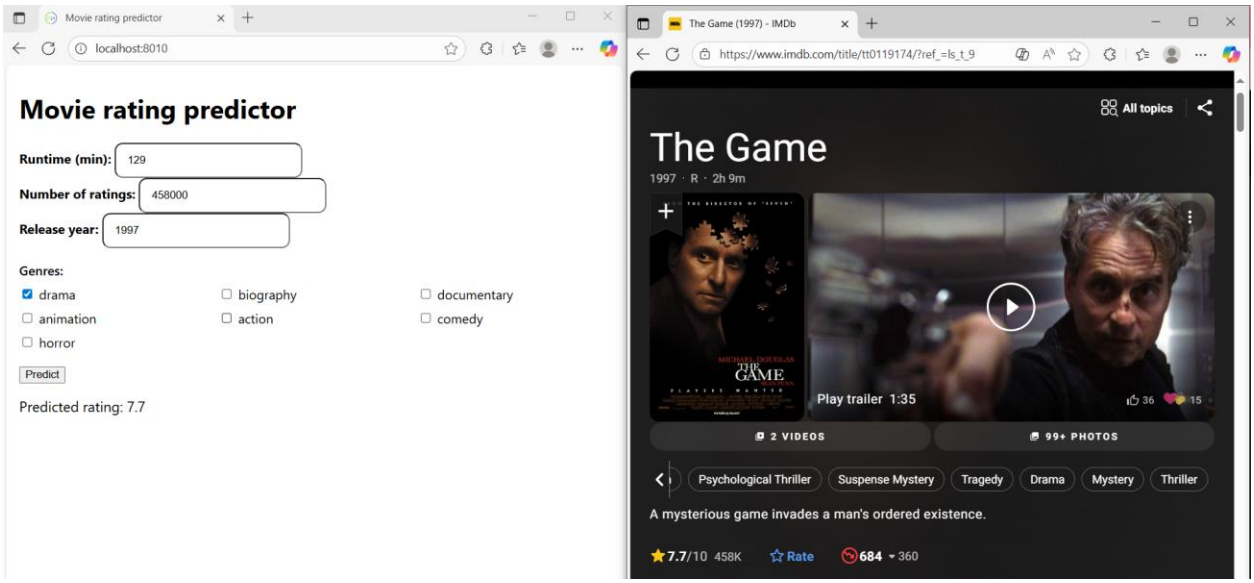
```
    ;;  (crit/bench
    ;;  (binding [*out* (java.io.StringWriter.)]
    ;;    (corr/print-multicollinearity data 0.8)))
  ; Evaluation count : 60 in 60 samples of 1 calls.
;             Execution time mean : 5.687586 sec
;     Execution time std-deviation : 303.625410 ms
;    Execution time lower quantile : 5.388505 sec ( 2.5%)
;    Execution time upper quantile : 6.369662 sec (97.5%)
;                    Overhead used : 10.399420 ns
    (crit/bench
   (binding [*out* (java.io.StringWriter.)]
     (corr/print-multicollinearity data 0.8 cfg/feature-columns)))
     ; Evaluation count : 180 in 60 samples of 3 calls.
;             Execution time mean : 432.367882 ms
;     Execution time std-deviation : 32.930956 ms
;    Execution time lower quantile : 397.138648 ms ( 2.5%)
;    Execution time upper quantile : 535.085186 ms (97.5%)
;                    Overhead used : 10.399420 ns
```

Slika 23 - Prikaz poboljšanja fje multicollinear-pairs

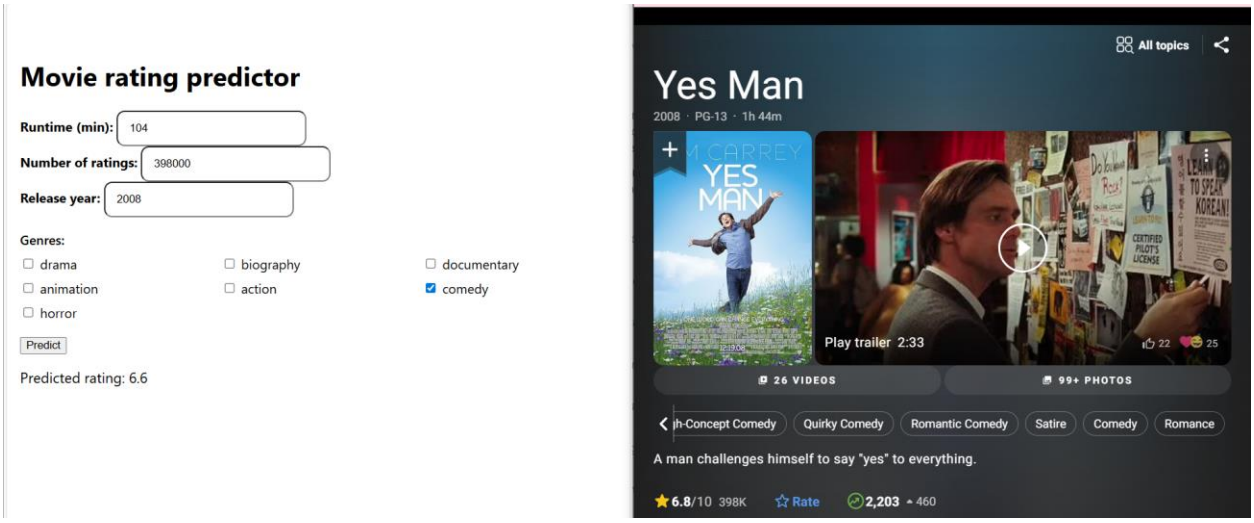
Na prvoj slici je početna implementacija multicollinear-pairs, na drugoj je optimizovana multicollinear-pairs,a na trećoj je prikazan rezultat merenja: pozivam corr/print-multicollinearity (koja suštinski delegira najskuplji deo posla na multicollinear-pairs), i prosečno vreme je palo sa ~5.69 s na ~0.4 s po pozivu. Ovo sam uradila jer sam primetila da print-multicollinearity traje dugo pri pozivu u core.clj.

Tokom benchmarka je korišćen (binding [*out* (java.io.StringWriter.)] ...) da u potpunosti utišam ispis zbog zatrpavanja REPL-a.
Glavne promene koje su omogućile desetostruko ubrzanje su: umesto da se za svaki par kolona iznova prave vektori vrednosti (xi, xj) prolaskom kroz ceo data (što radi stara funkcija), nova verzija jednom na početku prekompjutuje vektor za svaku traženu kolonu i čuva ga u mapi (col-v), pa u petlji nad parovima samo čita već spremne vektore i poziva stats/correlation. Druga bitna razlika je što funkcija radi nad **užim skupom** feature-columns (13 kolona) umesto nad svim kolonama (28), pa broj parova pada sa 378 na 78 (~4.8× manje posla).

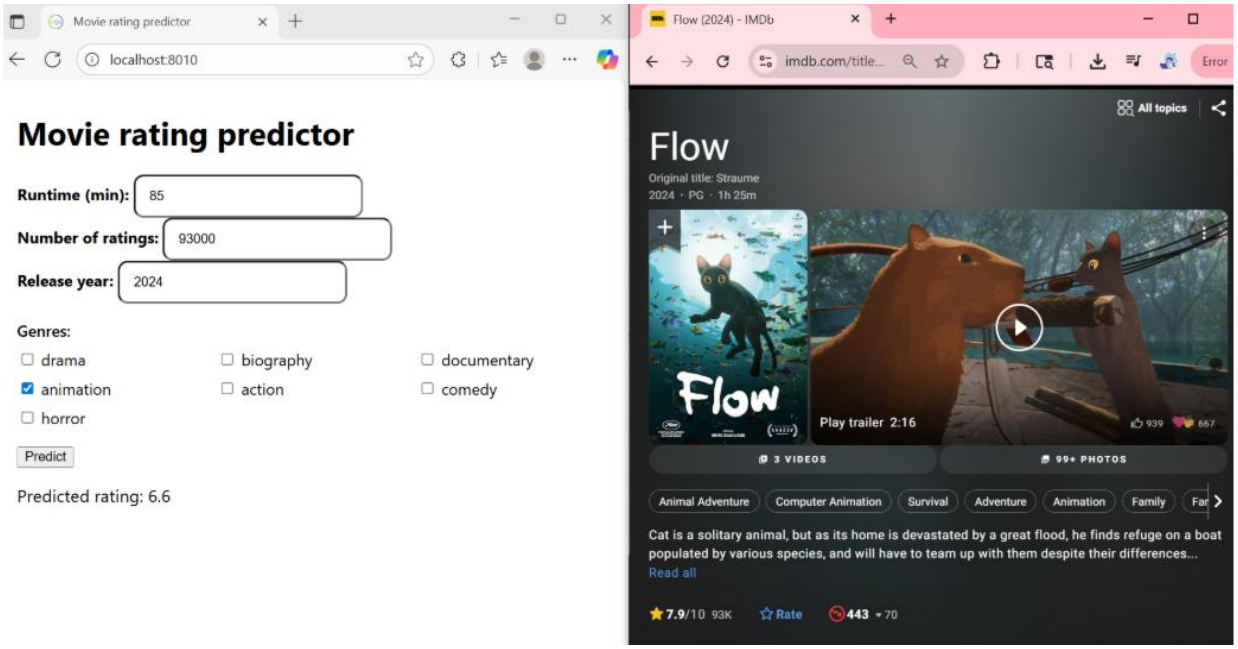
Primeri



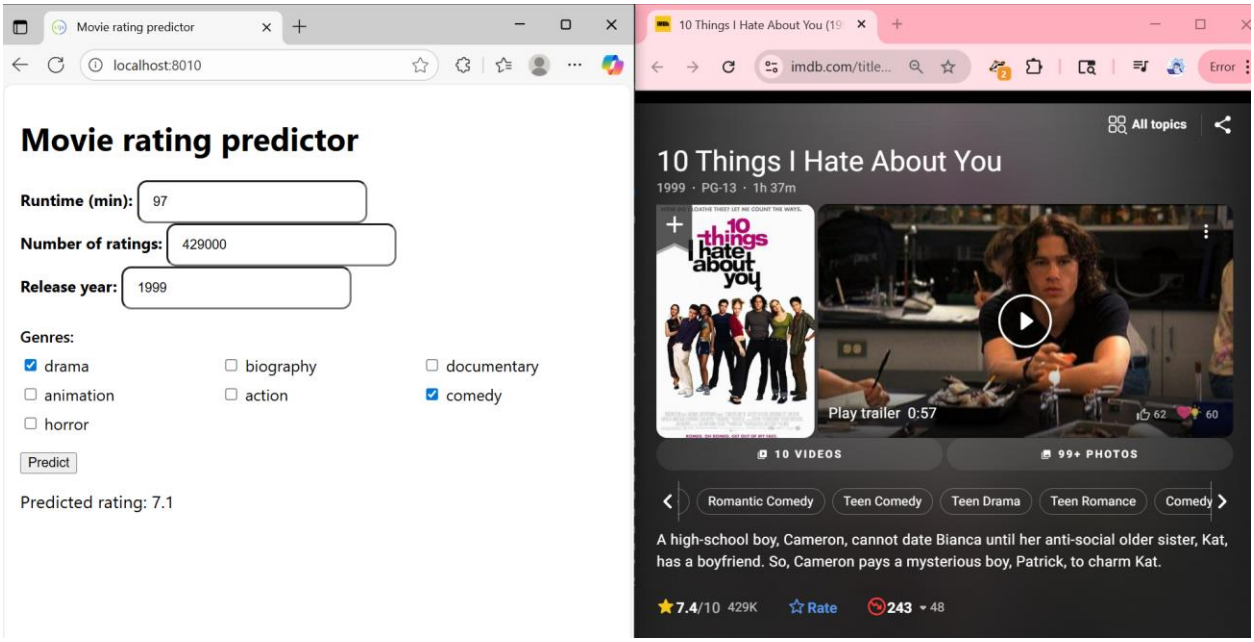
Slika 24 - Prikaz predviđanja ocene za film The Game



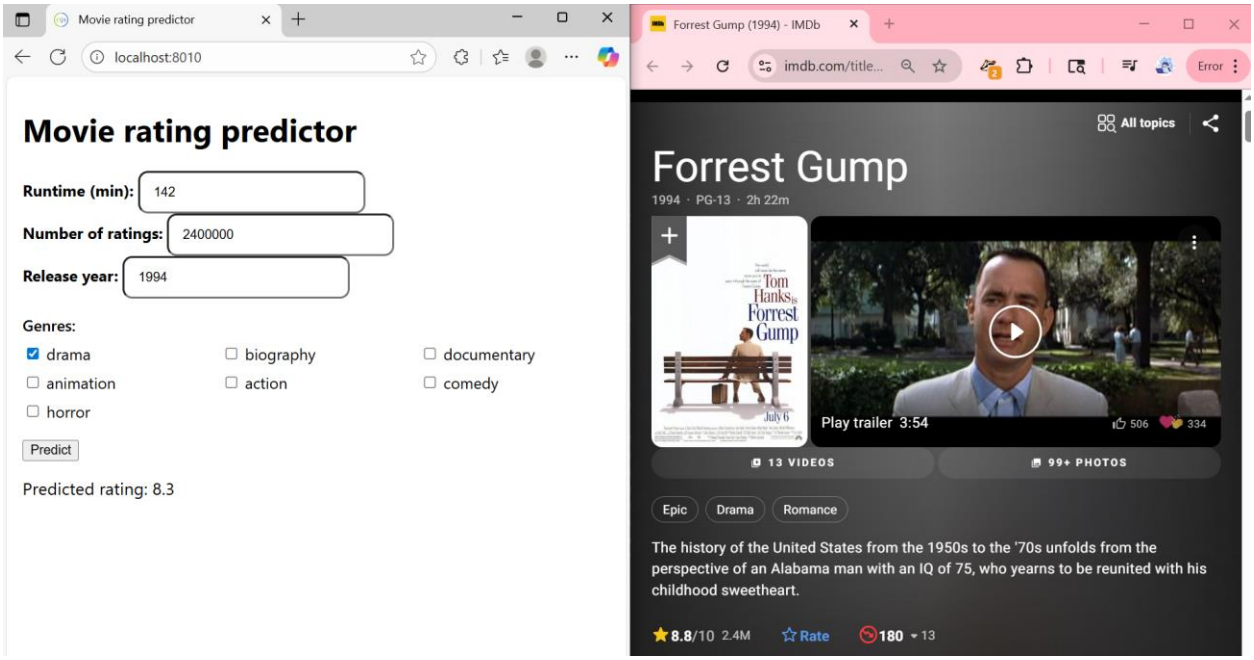
Slika 25 - Prikaz predviđanja ocene za film Yes Man



Slika 26 - Prikaz predviđanja ocene za film Flow

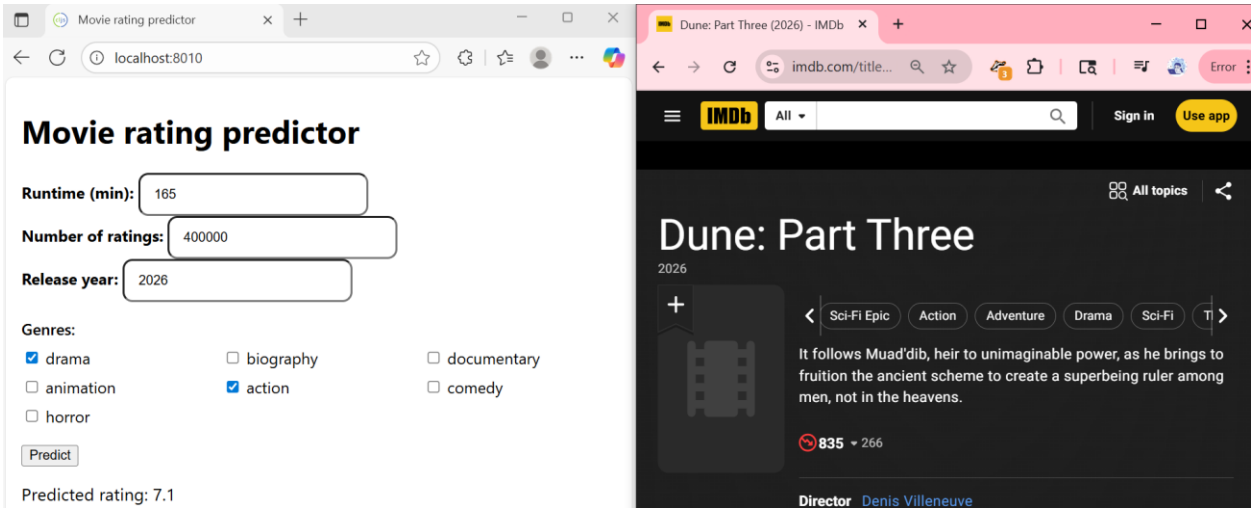


Slika 27 - Prikaz predviđanja ocene za film 10 Things I Hate About You



Slika 28 - Prikaz predviđanja ocene za film Forrest Gump

Zašto su neke procene tačne, a neke odstupaju?
Za deo filmova model „pogodi“ jer kombinacija osnovnih metapodataka (runtime, godina, žanrovi, broj glasova) daje stabilan signal. Kod drugih, razlika je mala (npr. 7.1 vs 7.4) - čest razlog je što **IMDb ne prikazuje tačan broj glasova već zaokružuje** (npr. *93k* umesto **92 736**). Pošto model nad brojem glasova radi log1p i standardizaciju, i mala razlika u ulazu može da pomeri predikciju za **0.2-0.3**. Dodatno, model koristi samo osnovne atribute (ne „vidi“ recenzije, glumačku ekipu, nagrade, hype), a ocene na IMDb-u su subjektivne i menjaju se tokom vremena - sve to može da dovede do sitnih odstupanja i kada je predikcija blizu.



Slika 29 - Prikaz predviđanja ocene filma koji još nema IMDb rating - Dune 3

Ako bismo sada pokušali da predvidimo ocenu filma koji još nema zvaničan IMDb rating, mogli bismo da napravimo okvirnu „preview“ procenu samo iz metapodataka.

Ako pretpostavimo **runtime** ~165 min, ~400k **glasova**, model (baziran samo na metapodacima) procenjuje ocenu oko **7.1**, ali uz tipičnu grešku ~±0.8 i uz napomenu da faktori koje ne vidi - recenzije, glumci, nagrade/“buzz” - mogu značajno pomeriti rezultat gore ili dole.