**SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA**

**Sveučilišni diplomski studij računarstva**

**Informacijske i podatkovne znanosti, DRD**

**PREDIKCIJA BROJA INSTALACIJA APLIKACIJE NA GOOGLE PLAY STORE-u**

**Računarstvo usluga i analiza podataka**

**Matej Dmitrović**

**Osijek, 2020.**

Sadržaj

[1. UVOD 1](#_Toc49441277)

[2. ANALIZA PODATAKA 2](#_Toc49441278)

[2.1 Skup podataka 2](#_Toc49441279)

[2.2 *Azure Machine Learning Classic* 2](#_Toc49441280)

[2.2.1 Priprema podataka 3](#_Toc49441281)

[2.2.2 Treniranje modela 5](#_Toc49441282)

[2.2.3 Azure Web service 6](#_Toc49441283)

[3. PROGRAMSKO RJEŠENJE 8](#_Toc49441284)

[3.1 Azure Machine Learning Classic 8](#_Toc49441285)

[3.2 HTML/CSS 8](#_Toc49441286)

[3.3 Python/Flask i jQuery 9](#_Toc49441287)

[3.3.1 „Denormalizacija“ 10](#_Toc49441288)

[3.3.2 API 10](#_Toc49441289)

[3.4 Primjer korištenja 11](#_Toc49441290)

[4. ZAKLJUČAK 13](#_Toc49441291)

[5. POVEZNICE I LITERATURA 14](#_Toc49441292)

# 1. UVOD

Mobilne aplikacije su jedno od najvećih tržišta modernog doba. Gotovo svaka osoba na svijetu posjeduje ili je posjedobala barem jedan pametni mobilni uređaj. Zbog velikog broja korisnika, na trgovinama za aplikacije poput Andridoving Google Play Store-a ili iPhoneovog App Store-a postoje aplikacije za svaku osobu. Neke aplikacije se prodaju više od drugih. Razvojem strojnog učenja otvorila se mogućnost analize tih aplikacija te na temelju raznih čimbenika predvidjeti prodaju aplikacije u razvoju. Upravu ta mogućnost se istražuje u ovom projektu.

# 2. ANALIZA PODATAKA

Kako bi bilo moguće učenje nad velikom količinom podataka, traženi podaci moraju biti dobro formatirani i uređeni kako učenje nad njima bilo što kvalitetnije. Također je bitno raspoznati koji podaci su bitni prilikom učenja te koji ne doprinose kranjem rezultatu.

## 2.1 Skup podataka

Skup podataka nad kojem je projekt rađen dohvaćen je sa stranice Kaggle koja drži repozitorij veliku količinu različitih skupova podataka. Odabran je skup podataka „Google Play Store“ koji u sebi sadrži mnoštvo aplikacija sa (nekim od) njihovih parametara. Na slici 2.1. prikazani su svi stupci koji su došli sa na obrađenim skupom podataka. Cijelokupni skup podataka sadrži približno 11 000 unosa.

* *App* – ime aplikacije
* *Category* – kategorija aplikacije
* *Rating* – prosješna ocjena aplikacije
* *Reviews* – broj recenzija za aplikaciju
* *Size* – veličina aplikacije (MB)
* *Installs* – broj instalacija aplikacije
* *Type* – je li aplikacija besplatna ili ne
* *Price* – cijena aplikacije
* *Content Rating* – ocjena sadržaja aplikacije po dobi
* *Genres* – žanr aplikacije, isto kao *Category*
* *Last Updated* – datum zadnjeg ažuriranja aplikacije
* *Current Ver* – trenutna verzija aplikacije
* *Android Ver* – verzija Androida za koju je aplikacija napravljena



Slika 2.1. Stupci korišteni u skupu podataka „Google Play Store“

## 2.2 *Azure Machine Learning Classic*

Strojno učenje u ovom projektu provodi se pomoću Microsoftove *Azure Machine Learning Classic* (u nastavku Azure ML) platforme za strojno učenje. Azure ML nudi stvaranje eksperimenata za strojno učenje koji se kasnije mogu razviti na Azureov web servis uslugu, kako bi vanjske stranice mogle koristiti stvoreni eksperiment. Azure ML u svojim eksperimentima nudi mnoštvo modula koji se mogu koristiti za pripremu podataka (čišćenje, normaliziranje, spajanje itd.), analizu teksta i strojno učenje (regresije, skupovi, klasifikacija i detekcija nepravilnosti). Postupak stvaranja modela je pojednostavljen time što se moduli mogu spajati klikom miša, tako da je izlaz jednog modula ulaz drugog. U suštiti, korisnik ne mora poznavati niti jedan programski jezik kako bi se koristio ovom platformom. Eksperiment se dijeli na eksperiment za treniranje, te kada je nađen pogodan model, stvara se predikcijski eksperiment. U predikcijskom eksperimentu moguće je odrediti koji (i koliko) su izlazi i ulazi za web-servis, te ukoliko je potrebno, odraditi još neke dodatne operacije nad podacima. Zato što trebabo odrediti točno jedan broj na temelju ostalih parametara, u ovom projektu razmatra se problem regresije.

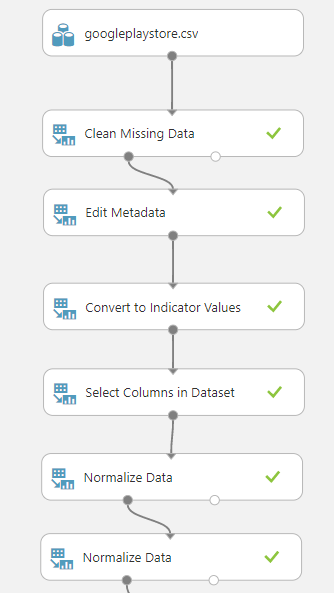
### 2.2.1 Priprema podataka

Dani skup podataka potrebno je pripremiti kako bi učenje nad njima bilo što kvalitetnije i rezultati što točniji. Azure ML nudi mnoštvo modula za manipuliranje podacima, no ne nudi sve potrebne za ovaj projekt. Dio pripreme podataka je proveden u Excel-u: maknuti su zarezi i znakovi plus kod svih brojčanih vrijednosti (broj instalacija i broj recenzija), te na mjestima gdje je bila oznaka „non“ za označavanje vrijednosti za koju nema podataka ostavljeno je prazno mjesto. Slijedeće pripreme su se odrađivalue u Azure ML-u. Na slici 2.2. nalazi se prikaz modula u Azure ML-u koji su zaslužni za pripremu podatka. Moduli, od gornjeg do doljnjeg, predstavljaju: skup podataka, čišćenje nedostajućih podataka, uređivanje metapodataka, pretvorba kategoričnih podataka, odabiranje stupaca i dva modula za normalizaciju podataka.

Prvi modul je datoteka sa podacima nad kojem će se provoditi učenje. Ti podaci su dodatno formatirani u Excel-u. Zatim se brišu redovi kojima nedostaju podaci. Nakon toga, uređuju se metapodaci, tj. stupovima se određuje kojeg su tipa i jesu li kategorični. Kategorični skupovi su oni koji predstavljaju neku kategoriju – u ovom slučaju to su stupovi *Category* i *Content Rating*. Ovaj korak je nužan ukoliko će se kasnije u eksperimentu kategorični podaci prebaciti u numeričke – što je i sljedeći korak. Kategorični podaci se dijele na više stupaca čija vrijednost je ili 0 ili 1. Svaka kategorija ima svoj stupac. U ovakvom pristupu, sve kategorije prilikom učenja imaju jednaku važnost, za razliku ako bi se kategorije pridruživale

*kategorija0 = 0, kategorija1 = 1, … , kategorijan = n*

jer bi tada *kategorijan* imala veću važnost od ostalih kategorija. Nakon svega navedenog, odabiru se stupci koji će se koristiti pilikom strojnog učenja. Stupci koji nisu odabrani su *App, Size, Type, Genres, Last Updated, Current Ver* i *Android Ver*. Ime aplikacije, veličina, datum zanjeg ažuriranja, trenutna verzija aplikacije i verzija Androida za koju je aplikacija pravljena ne utječu na broj instalacija. Tip aplikacija predstavlja je li aplikacija besplatana ili ne, što se već može očitati iz cijene, dok žanr aplikacije predstavlja njenu kategoriju, ali u ljudima čitkijem zapisu.

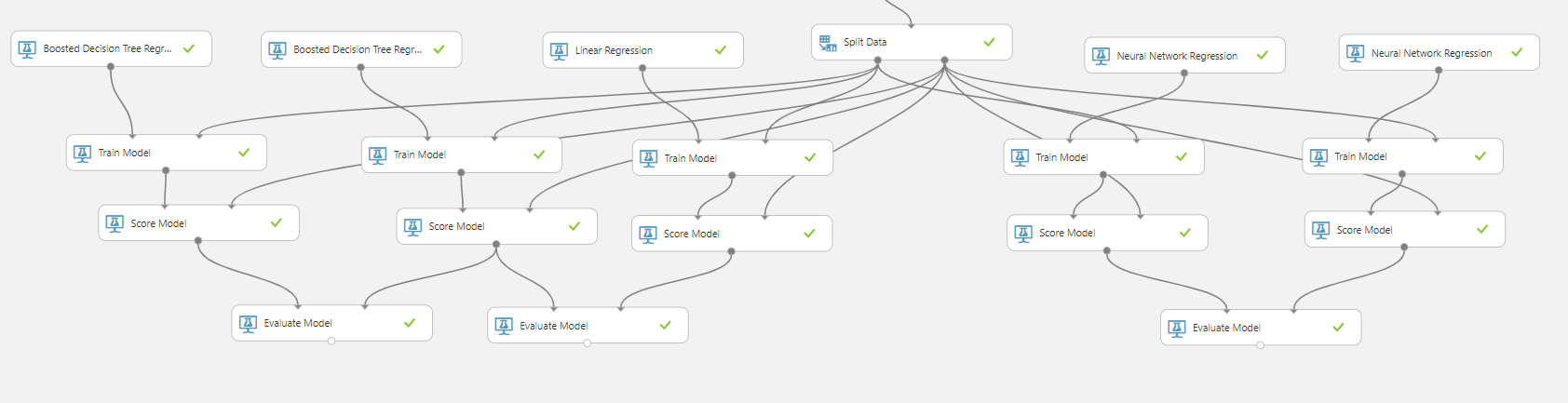


Slika 2.2. Moduli za pripremu skupa podataka u Azure ML-u.

Naposlijetku slijede normalizacje podataka. Za to su dva različita modula jer se vrše dvije različite normalizacije nad različitim stupcima. Prva normalizacija se vrši nad prosječnom ocjenom. Koristi se *Min-Max* metoda koja za svaku vrijednost stavlja u interval [0, 1] na temelju najmanje i najveće vrijednosti stupca. Ova metda je pogodna za taj stupac jer vrijednosti su iz intervala [1, 5] te svaka vrijednost nakon normalizacije ima istu važnost kao i prije nje. Druga normalizacija se vrši nad stupcima za broj instalacija, broj recenzija i cijene aplikacije. Koristi se *LogNormal* metoda zbog velikih razlika između najmanjih i najvećih vrijednosti stupaca. *LogNormal* koristi normalnu distribuciju i logaritmiranje kako normalizirani podaci bili lognormalno distribuirani u intervalu [0, 1] te tako manje vrijednsoti imaju veće značenje nego kada bi se koristila bilo koja druga metoda normalizacije.

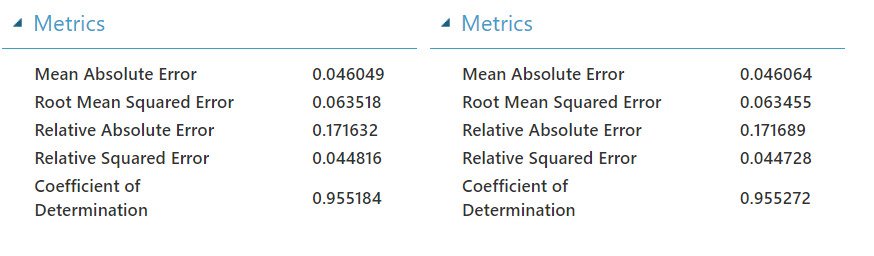
### 2.2.2 Treniranje modela

Nakon pripreme podataka, potrebno je vrštiti treniranje nad njima. Pripremljeni podaci se dijele na podatke sa učenje i podatke za testiranje pomoću modula *Split Data*. Podaci čenje i testiranje se dijelle u omjeru 3 : 1. Slika 2.3. prikazuje cijelokupni graf modula koji su korišteni za učenje nad podacima. Zbog što pokušavamo predvidjeti numeričku vrijednost na temelju ostalih podataka, obrađuje regresijski problem. Određeno je pet modela za strojno učenje za regresije: dva pojačana stabla odluka, dvije neuronske mreže i jedna linearna regresija. Svaki model ima različite parametre. Nakon treniranja svakog modela, model se ocjenjuje na temelju testnog skupa podataka. Na kraju se modeli uspoređuju s drugim modelima kako bi bilo moguće odabrati model koji najbolje odgovara ovom problemu.

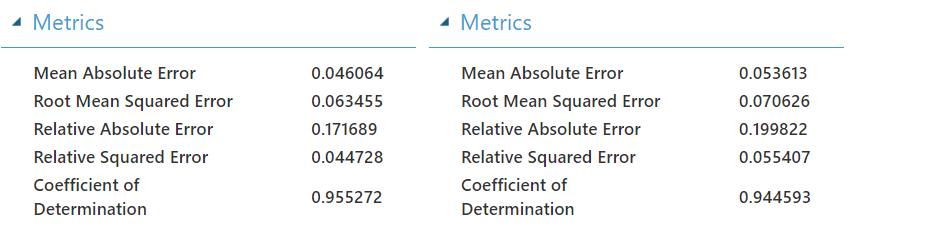


Slika 2.3. Cijelokupni prikaz dijela eksperimenta za učenje nad podaci.

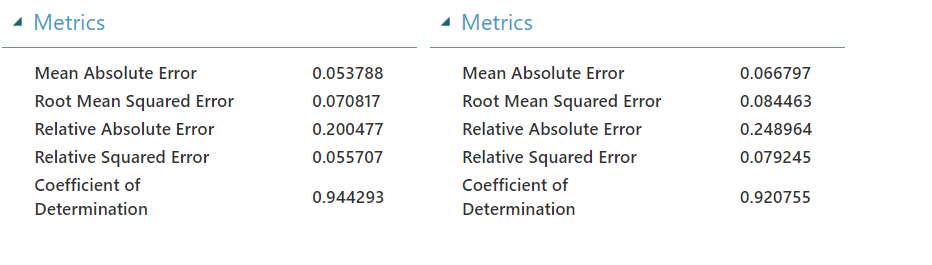
Slika 2.4. – 2.6. prikazuju usporedbe rezultata među navedenim modelima. Parametar koji je najbitniji za odabir modela jest koeficijent deteminacije. Ukoliko se promatra taj parametar, neuronske mreže su imale najgore rezultate dok su pojačana stabla imala najbolji, tako da je za model strojnog učenja odabran jedan od stabala. Odabrano stablo, lijevo sa slike 2.4., ima malo manji koeficijent determinacije od drugog, ali ima ponešto manju apsolutnu i relativno pogrešku, tako da ako greška postoji, bit će manja nego kod drugog stabla. Odabrano stablo ima najviše 30 listova po stablu, minimalno 10 nizova treniranja da se formira list, 0.1 ratu učenja i sveukupno 100 stabala napravljeno.



Slika 2.4. Razlika rezultata dva pojačana stabala odluka.



Slika 2.5. Razlika rezultata između jednog od stabala odluke i linearne regresije.

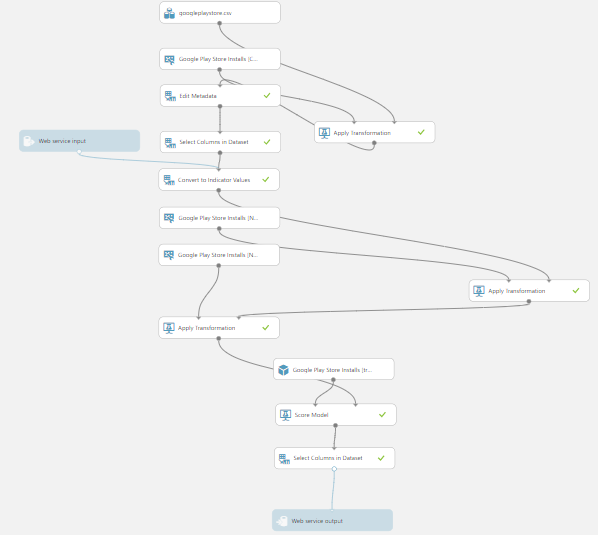


2.6. Ralika rezultata neuronskih mreža.

### 2.2.3 Azure Web service

Nakon odabira modela, željeni model se prebacuje u predikcijski eksperiment. Na slici 2.7. prikazan je cijeli graf koji nastane za predikcijski eksperiment. U tom grafu, najvažniji moduli su *Web Service Input* i *Web Service Output* koji služe za određivanje ulaza i izlaza web-usluge. Ulaz se spaja na modul za pretvorbu kategoričnih podataka kako bi ulazi podaci bili formatorani kao kategorije. Budući da je korisniku potrebna samo vrijednost predikcije, odabire se samo stupac *Scored Labels* koji upravo to i označava. Vrijednosti tog stupca se šalju nakon upita na API web-servisa.

Nakon razvoja predikcijskog eksperimenta na web-servis, Azure ML nudi API ključ i primjere poziva API-ja kako bi se mogla ostvariti cijelokupna API usluga. Također je ponuđen format u kojem podaci moraju biti poslani kako bi se izvršila predikcija nad njima. Radi pojednostavljenja tih poziva, web-stranica projekta ima svoj vlastiti API.



Slika 2.7. Prikaz cijelog grafa za predikcijski eksperiment.

# 3. PROGRAMSKO RJEŠENJE

## 3.1 Azure Machine Learning Classic

Azure ML unutar svog web servsa nudi primjer pozivanja API-ja kao i detalje kako se API korsiti. Za dovaćanje predikcija je korišten Python programski jezik koji je uklopnjen u sklop web-stranice pomoću Flask framework-a.

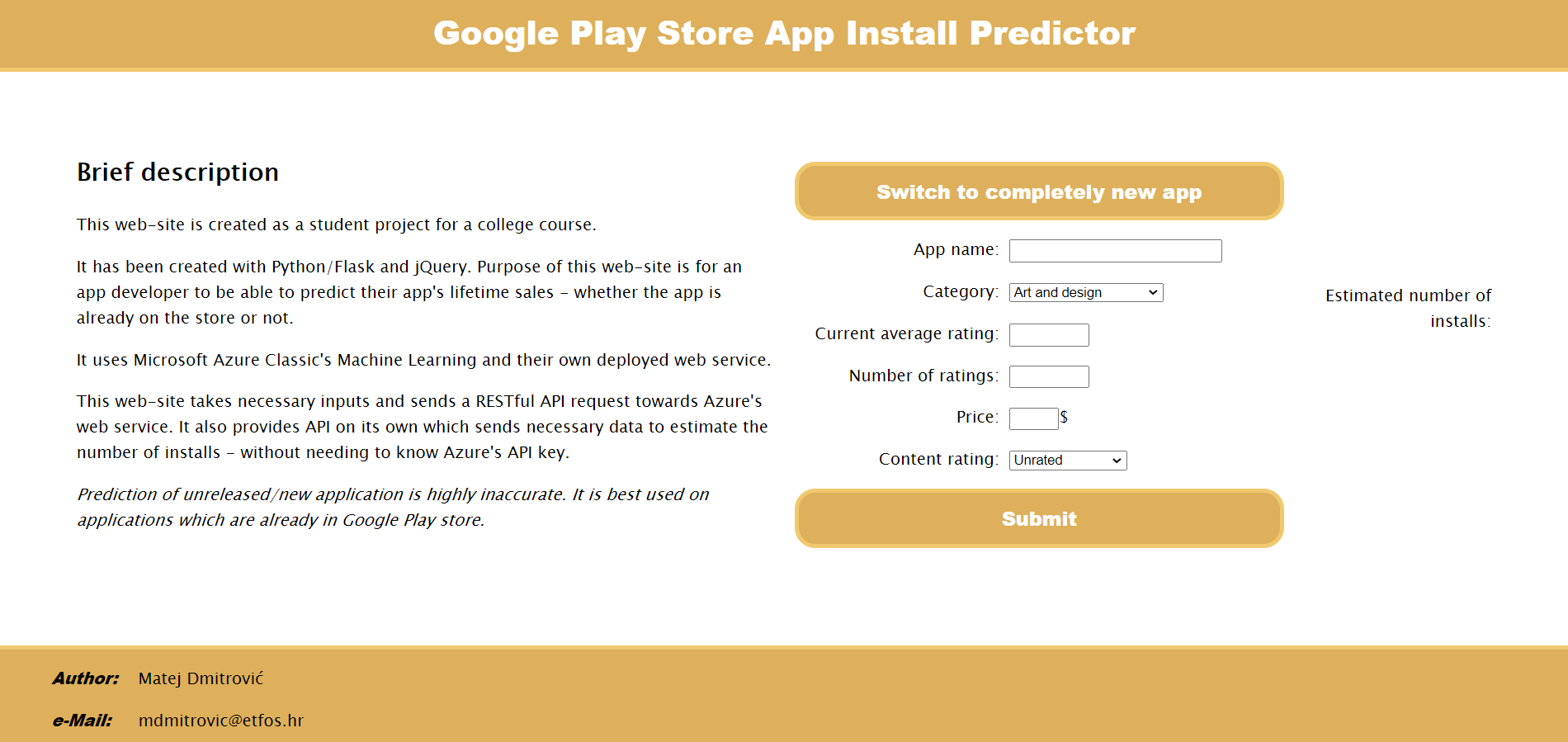
## 3.2 HTML/CSS

Projekt se sastoji od jedna HTML stranice u kojoj se nalazi forma za upisivanje parametara predikcije. Forma se sastoji od ulaza za:

* Ime aplikacije – tekstualno polje za ime aplikacije
* Kategorija – padajući izbornik za odabir kategorija aplikacije. Moguće opcijuesu dohvaćene sa API-ja Azure web servisa
* Prosječna ocjena – numeričko polje za prosječnu ocjenu recenzija aplikacije, ukoliko je aplikacija već na Google Play Store-u
* Broj recenzija – numeričko polje za broj recenzija aplikacije, ukoliko je aplikacija već na Google Play Store-u
* Cijena – numeričko polje za cijenu aplikacije
* Ocjena sadržaja – padajući izbornik za odabir ocjenu sadržaja po dobi. Moguće opcije su dohvaćane sa API-ja Azure web servisa

„Kategorija“ i „Ocjena sadržaja“ su dinamički popunjeni pomoću jQuery-a kako bi se stranica mogla lakše ažurirati ako dođe do promjena vrijednosti koja ta polja mogu poprimiti. Na slici 3.1. prikazan je izgled cijele web-stranice. HTML-om je definirana forma elemenata i njihov raspored unutar iste razine. Glavna raspodjela elemenata organizirana je na zaglavlje, tijelo stranice i podnožje. Zato što je projekt sastavljen od samo jedna web-stranice, nisu korišteni navigacijska traka/meni i bočna traka.

Nakon definiranja strukture, CSS-om se određuje stil stranice. Većina stvari koje su definirane CSS-om su boje elemenata te njihove margine. Također je definirano linijsko poravnavanje podjelnjih elemenata *<div>* kako bi bili sa strane jedno pored drugog. Zbog limitacija takvog prustupa, dio za izricanje predikcije izgleda oskudno.



Slika 3.1. Prikaz cijele web-stranice za predikciju broja instalacija aplikacije na Google Play Store-u.

## 3.3 Python/Flask i jQuery

Dinamični dio web-stranice napravljen je pomoću Flask framework-a za Python i jQuery-a. Flask održava organizaciju web-stranica i API poziva te ima mogućnost za podizanje vritualnog servera kako bi stranica mogla pristupiti vanjskim podacima bez podizanja stranice na pravi server. Temelji se na Python programskom jeziku, te je jedan od razloga korištenja Python-a za dohvaćanja predikacije sa Azure web servisa. Cijela dinamičnost web stranice se temelji na tome da korisnik upisuje ulaze u polja za formu, te klikom na gumb submit poziva se jQuery funkcija koja šalje podatke forme podatke na API ove web-stranice. Flask tada preuzima podatke i šalje ih u Python kod za slanje podataka Azure web servisu. Ukoliko ne dođe odgovor ili je došlo do greške, Python će vratiti grešku u obliku stringa. Ako je sve u redu i predikcija je bila uspješna, Python će formatirati i „denormalizirati“ podatke te ih postali natrag jQuery-u. jQuery tada predočuje dobivenu predikciju ili grešku na određeno mjestu unutar web-stranice.

Korisnik također ima mogućnost odabiranja predikciju za novu aplikacije bez recenzija klikom na gornji gumb. No, zbog važnosti tog parametra, predikcije su najčešće netočne.

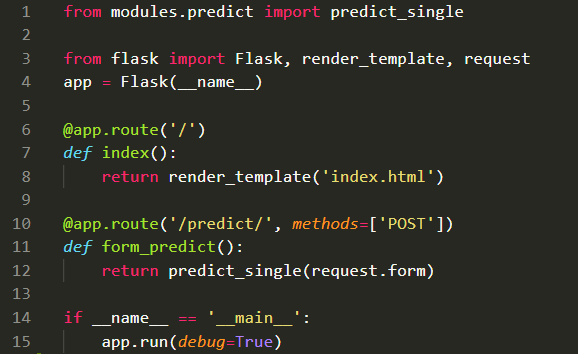
### 3.3.1 „Denormalizacija“

U ovom pod-poglavlju spomenut je pojam „denormalizacije“ – pretvorbe dobivene predikcije natrag u odgovarajući interval vrijednosti. Predikcija kao rezultat normalizacije ostalih elemnata prilikom strojnog učenja dolazi u intervalu [0, 1]. Taj interval nije dobar pokazivač broja instalacija neke aplikacije. Budući da je u trening dijelu strojnog učenja provedena LogNormal normalizacija stupca „Installs“, predikcija se mora provesti kroz inverznu LogNormal funkciju. Azure ML ne prikazuje detalje normalizacija stoga je inverzija LogNormal normalizacije teško. U tom slučaju koristi se aproksimacija inverzne funkcije koja se temelji na empirijskim vrijednostima. Dobivena aproksimacija funkcija ima izraz

gdje je x dobivena vrijednost predikcije a y konkretan broj predikcije instalacija aplikacije.

### 3.3.2 API

API je jednostavno definiran pomoću Flask frameworka – potrebno je naglasiti na koji link se poziva API te pomoću kojih metoda. Slika 3.2. prikazuje programski kod za stvaranje cijelokupnog API-ja web-stranice. Na toj slici se može primjetiti, ako se pozove link „/predict/“ na web-stranicu sa 'POST' metodom, da će API vratiti vrijednost predikcije jedne forma. U načelu, poziva se Python funkcija definirana u drugoj datoteci koja upravo svrhu slanja, dohvaćanja i „denormalizacije“ predikcije.

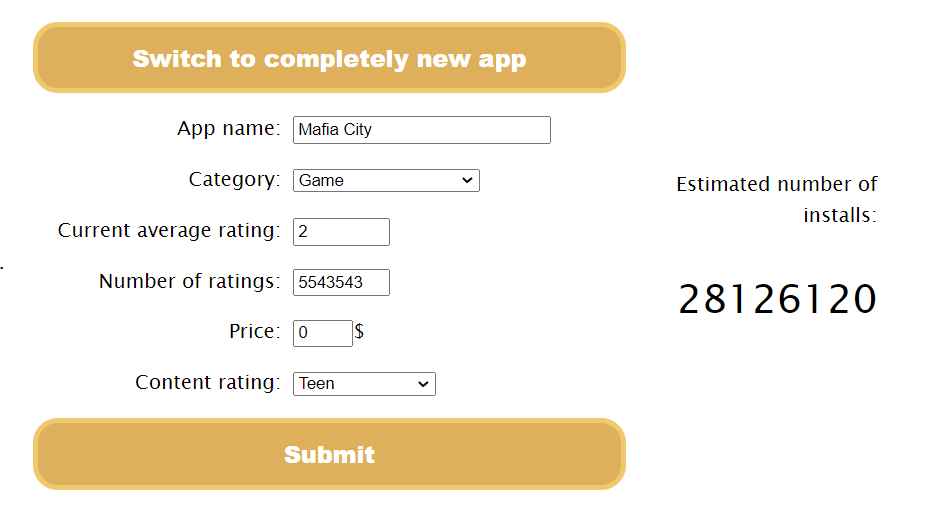


Slika 3.2. Programski kod za kreiranje API-ja.

Ukoliko je potrebno dodati još API poziva, u navedni kod se analogno mogu dodati ostali pozivi, tako da je nadograđivanje sustava jednostavno.

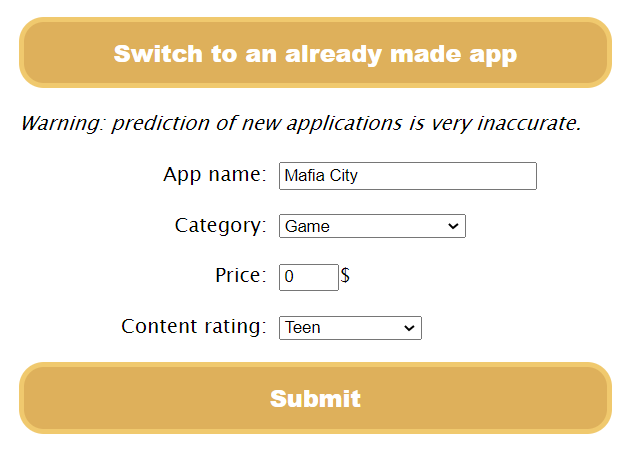
## 3.4 Primjer korištenja

Web-stranica se koristi tako što se u formu upisuju tražene vrijednosti. Slika 3.3. prikazuje jedan primjer forme i rezultata koji se dobije za unešene parametre.



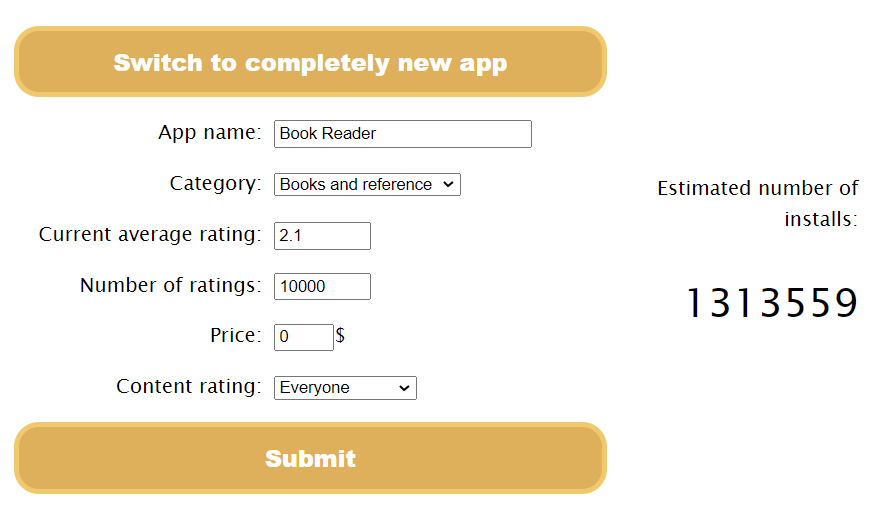
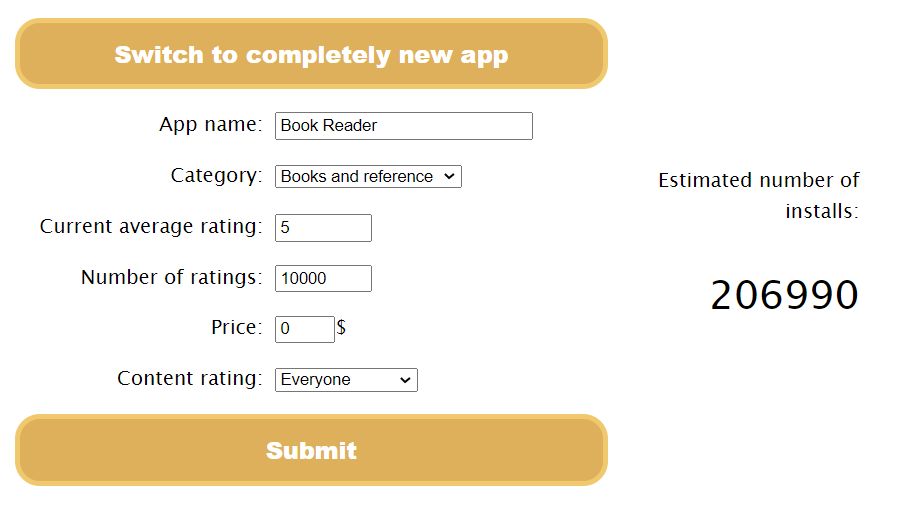
Slika 3.3. Primjer korištenja forme.

Ukoliko se pritisne gumb „Switch to completely new app“ dobije se skraćena forma sa slike 3.4. U ovoj formi se ne unose parametri vezani uz recenziju aplikacije budući da se samtra da aplikacija još nije postavljena na Google Play Store.



Slika 3.4. Skraćena forma za predikciju broja instalacija.

Naučeni model je primjetio neke značajke koje možda nisu intuitivne, poput toga da aplikacija s manjom srednjom ocjenom ima veći broj instalacija. Taj primjer je prikazan na slici 3.5. gdje su predstavljene predikcije za dvije aplikacije: jedna s ocjenom 5 i druga s ocjenom 2.1; svi ostali parametri su isti. Broj instalacije aplikacije s boljom ocjenom (5) je gotovo šest puno manji nego broj instalacija aplikacija s lošijom ocjenom (2.1). To se može tumačiti kako aplikacije koje su loše imaju veći broj instalacija zbog svojih reklama ili lažnog oglašavanje te su recenzije zbog toga lošije dok je broj instalacija velik.



Slika 3.5. Usporedba broja intalacija između aplikacije s boljom ocjenom (lijevo) i aplikacije s lošijom ocjenom (desno).

# 4. ZAKLJUČAK

Strojno učenje daje nam novi način da istražujemo i analiziramo nama već poznate stvari budući da strojevi mogu uočiti neke korelacije koje mi ne možemo. No, kako bi strojno učenje moglo biti moguće, skup podataka nad kojem će učenje biti vršeno mora biti dobro pripremljen inače rezultati mogu divergirati od stvarnih vrijednosti. U ovom projektu pokazala se priprema podataka i učenje nad njima, te stvaranje vanjskog sustava koji koristi naučeni model kako bi se stvorila usluga koji drugi ljudi mogu koristiti. Iako ovaj projekt možda nije za neku ozbilju upotrebu, on služi kao dokaz da je moguće pronaći neke veze za koje nismo ni bili svjesni.

# 5. POVEZNICE I LITERATURA

Programsko rješenje zajedno sa ovom dokumentacijom se nalazi na udaljenom GitHub repozitoriju koji se može pristupiti preko linka:

* <https://github.com/Matt-Tyrantin/RUAP-project>.

Direktan zahtjev na Azure ML servis se može poslati na:

* <https://ussouthcentral.services.azureml.net/workspaces/92518c6337964d95af1c417d83471dec/services/535fd871d310474ea32a096456a43822/execute?api-version=2.0&details=true>

, sa API ključem:

* OkBFV/GW3n/rfqOXChEb+/yp3oe/s7pPa0UzKm0tW6dJAau16MWuou28WKI+KSZxpUGfD6tgwHhxlWRl4WeCoA==

Flask dokumentacija se može proučiti na:

* <https://flask.palletsprojects.com/en/1.1.x/>

Skup podataka „Google Play Store“ je preuzet sa:

* <https://www.kaggle.com/datasets>