Predviđanje cene artikla - Mercari

# Uvod

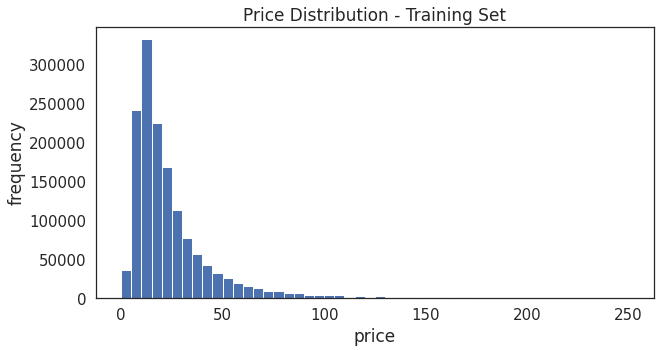
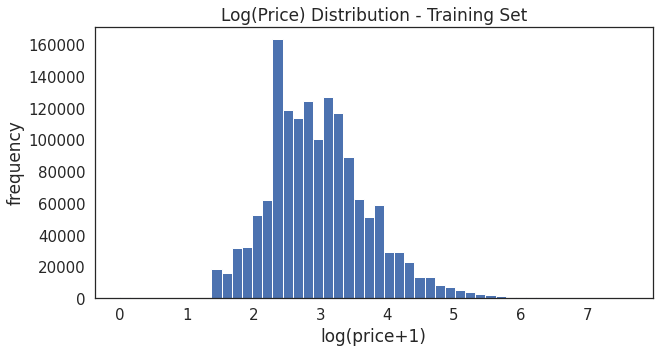
Utvrditi koliko nešto zaista vredi može biti veoma teško. Cena nekog proizvoda zavisi od mnogo faktora, gde mali detalji mogu imati veliki uticaj. Određivanje cena proizvoda postaje još teže uzimanjem u obzir ogromne količine artikala koji se prodaju na internetu. Odeća na primer ima jake sezonske trendove i na nju u velikoj meri utiču nazivi brendova, dok cene elektronike variraju na osnovu specifikacija proizvoda. Predviđanje cene na osnovu nekoliko faktora prodavcima omogućava da lakše plasiraju svoj proizvod na tržište, bez potrebe za prethodnom analizom cena konkurencije. Ovaj postupak ima brojne benefite kao što su ušteda vremena i drugih resursa u istraživanje tržišta, kao i određivanje precizne cene bez rizika za gubitkom u slučaju postavljanja preniske cene ili za lošim odzivom kupaca u slučaju postavljanja previsoke cene.

U ovom radu biće opisan proces kreiranja modela na osnovu nekoliko faktora, vrši precizno predviđanje cene proizvoda. Taj proces će se sastojati od analize skupa podataka, pripreme skupa podataka, kreiranja i optimizacije više modela i izbora modela sa najboljim performansama i njegove primene. Za učenje modela korišćeni su podaci iz takmičenja „Mercari Price Suggestion Challenge“ organizovanog 2017. godine na Kaggle platformi.

# Analiza skupa podataka

Skup podataka sastoji se od oko 1,500,000 listinga proizvoda sa sledećim atributima: *train\_id* – id listinga; *name* – naslov listinga; *item\_condition\_id* – stanje proizvoda koje je obezbedio prodavac; *category\_name* – kategorija kojoj proizvod pripada; *brand\_name* – naziv brenda; *price* – cena po kojoj je proizvod prodat, ovo je atribut koji ćemo predviđati; *shipping* – 1 ukoliko dostavu plaća prodavac, 0 ukoliko dostavu plaća kupac i *item\_description* – opis proizvoda.

Prikazom raspodele izlazne promenljive *price* na dijagramu primećujemo da je ona nagnuta na desno i zbog toga nad ovom promenljivom vršimo logaritamsku transformaciju kako bismo normalizovali raspodelu, što je prikazano na slici ispod.

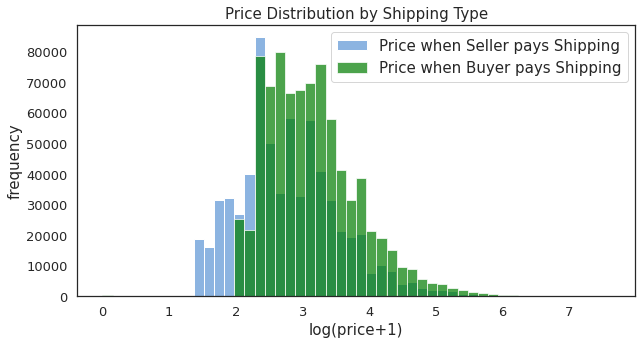
 

Slika 1 Raspodela cene pre transformacije (levo) i posle transformacije (desno)

Analizom atributa *shipping* primećujemo da u 45% listinga transport plaća prodavac dok u ostalih 55% listinga transport plaća kupac, što je prikazano na dijagramu dole.

Slika 2 Odnos vrednosti promenljive *shipping*

Razlaganjem raspodele cene na listinge kod kojih transport plaća kupac i listinge kod kojih transport plaća prodavac primećujemo da kod listinga sa manjom prodajnom cenom uglavnom prodavac plaća transport, dok je suprotan slučaj kod listinga čija je prodajna cena viša, što se može videti na dijagramu ispod.



Slika 3 Raspodela cene po tipu transporta

Atribut *category\_name* se sastoji od 1,287 jedinstvenih vrednosti i predstavlja niz karaktera koji sadrži glavnu kategoriju, potkategoriju prvog nivoa i potkategoriju drugog nivoa koje su međusobno odvojene kosom crtom. Atribut *brand\_name* sadrži 4,809 jedinstvenih naziva brenda. Primećujemo i da ovaj atribut ima veliki broj nedostajućih vrednosti: oko 630,000. Među najpopularnijim brendovima nalaze se PINK, Nike, Victoria’s Secret, Apple i drugi Atribut *item\_description* predstavlja opis proizvoda od strane kupca.

# Priprema podataka

Pre kreiranja modela izvršena je priprema podataka. Pre svega su otklonjene nedostajuće vrednosti iz atributa *brand\_name* zamenom vrednosti sa *Missing* i izbačena su četiri listinga bez opisa proizvoda. S obzirom na to da atribut kategorija u sebi sadrži tri nivoa kategorija odvojenih kosom crtom, one je podeljen na tri nova atributa: *general\_cat, subcat\_1* i *subcat\_2.* Ova tri nivoa kategorija su od velikog značaja jer omogućavaju preciznije grupisanje listinga što rezultira većom preciznošću modela i povećava moć generalizacije. Ovi atributi zajedno sa atributom *item\_condition­­\_id* pretvoreni u kategoričke atribute, radi uštede memorije.

Nad atributom *name*, kao i prethodno kreiranim atributima kategorija,primenjen je *CountVectorizer* koji generiše matricu broja tokena. Parametar *min\_df* je postavljen na vrednost 10, što znači da će se kolone matrice biti tokeni koji se u dokumentu pojavljuju barem 10 puta, a vrednosti u matrici biće frekvencije pojavljivanja pojedinačnih tokena u svakom listingu.

Nad atributom *item\_description* primenjen je TfidfVectorizer koji slično kao CountVectorizer generiše matricu tokena, ali su vrednosti u matrici proizvod frekvencije tokena u jednom dokumentu i inverzne frekvencije tokena u svim dokumentima. TfidfVectorizer je tehnika koja omogućava identifikaciju najvažnijih reči merenjem učestalosti pojavljivanja svake reči i koliko je ta reč jedinstvena u svim dokumentima. Ovo omogućava identifikovanje kategorije dokumenta što povećava prediktivnu moć modela.

Sledeći korak u pripremi je generisanje matrice naziva brendova korišćenjem *LabelBinarizer* klase koja binarizuje labele metodom one-vs-all. Nad atributima *item\_condition\_id* i ­*shipping* primenjena je funkcija *get\_dummies* koja binarizuje kategoričke atribute. Ove metode binarizacije su primenjene kako bi kategoriče atribute pretvorili u numeričke kojij su prikladni za algoritme regresije koji će biti korišćeni.

Na kraju su transformisani atributi spojeni u jedan skup koji sadrži 63,462 atributa. 90% podataka izdvojeno za trening modela a ostalih 10% za testiranje.

# Učenje i evaluacija modela

Algoritmi korišćeni za učenje modela su linearna regresija, ridge regresija i random forest regresor. Metrike korišćene za evaluaciju modela su R2 score i prethodno napravljen RMSLE score čija formula se nalazi ispod.

Model linearne regresije kreiran je sa podrazumevanim parametrima. Prilikom kreiranja modela ridge regresije, korišćena je *grid search* pretraga optimalnog parametra alfa, sa 3 kros validacije. Potencijalne vrednosti iznosile su 0.1, 0.5, 1, 2 i 3. Ove vrednosti su izabrane jer su se u praksi pokazale kao najbolje. Ovim postupkom utvrđena je optimalna vrednost koja iznosi alpha = 1. Parametar solver postavljen je na vrednost „auto“ koji omogućava algoritmu da automatski podesi solver koji je optimalan za dati ulaz. Parametar fit\_intercept postavljen je na True što je najčešći slučaj u praksi.

Random forest regresor sastoji se od 100 stabala što je predefinisana vrednost algoritma, dok je parametar maksimalne dubine stabla 6. Zbog ograničenih resursa, nije postojala mogućnost optimizacije parametara ovog modela, te su izabrani parametri koji su u praksi davali najbolje rezultate.

Metrike modela na test setu prikazane su u tabeli ispod.

Tabela 1 Metrike modela

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Vreme treniranja | R2 score | RMSLE score |
| Linearna regresija | 25m | **0.4827** | **0.4712** |
| Ridge regresija | **42s** | 0.4482 | 0.4719 |
| Random forest regresor | 59m | 0.1029 | 0.6676 |

# Zaključak i budući pravci

Na osnovu iznetih rezultata zaključujemo da model linearne regresije ima najbolje performanse sa R2 skorom 0.4827i RMSLE skorom 0.4712. Model ridge regresije je, međutim, pored sličnog RMSLE skora i R2 skora, imao značajno kraće vreme treniranja što je bitno napomenuti.

Što se tiče budućeg pravca, postoji nekoliko strategija koje bi mogle da poboljšaju preciznost. Izdvajanje većeg broja karakteristika pri korišćenju *CountVectorizer* i *TfidfVectorizer* klasa u pripremi podataka uticalo bi na poboljšanje preciznosti jer bi modeli učenje vršili nad većim brojem faktora. Međutim,najveći pomak u smanjenju greške ostvario bi se kreiranjem kompleksnijeg ansambl modela, kao što je na primer gradient boosting regresor i optimizacije njegovih hiperparametara. Ove metode nisu korišćene zbog hardverskih ograničenja.