# Introduction to Data Mining



## 3. domača naloga: napovedovanje prihodov avtobusov LPP

Na voljo imate podatke o vožnjah avtobusov LPP v letu 2012 od začetka januarja do konca novembra. Cilj naloge je zgraditi čim boljši model, ki bo znal iz časa odhoda z začetne postaje napovedati čas prihoda na končno postajo. Vaš model bomo preizkusili na podatkih o avtobusnih vožnjah v decembru 2012.

Predvidevamo, da boste za vsako avtobusno linijo zgradili svoj napovedni model. Značilk je trenutno zelo malo, poleg tega pa niso primerne za linearno regresijo. Za napovedovanje nujno uporabite linearno ali polinomsko regresijo. Eksperimentirate z različnimi tehnikami tvorbe predobdelave podatkov in tvorbe značilk.

Za dodatna navodila, primere, podatke in oddajo napovedi obiščite stran tekmovanja. Geslo smo vam poslali na elektronski naslov, s katerim ste se prijavili na učilnico. Če ga ne najdete, pišite na marko.toplak@fri.uni-lj.si.

**Predtekmovanje (obvezno)**. S strani predtekmovanja snemite podatke. Ti podatki opisujejo le eno avtobusno linijo (tekmovanje jih zajema več). Zanjo z linearno regresijo izdelajte napovedi in jih do vključno **18. 11. 2017** oddajte na lestvico. Vaš cilj je doseči rezultat pod 160.

**Ocenjevanje**:  50% prilagoditev značilk, da bodo koristile linearni regresiji in uporaba linearne regresije. Nadaljnjih 20% je vredna vaša implementacija internega preverjanja na učnih podatkih, 10% pravočasen predtekmovalni rezultat do 160, ostalih 20% pa bo izhajalo iz kakovosti implementiranih tehnik.

Bonusi:

- 1. mesto: ocena +50%.
- 2.-5. mesto: ocena +35%.
- 6.-10. mesto: ocena domače naloge +25%.
- 11.-15. mesto: ocena domače naloge + 15%
- 15.-20. mesto: ocena domače naloge + 10%
- 1. mesto na predtekmovanju: ocena +20%
- 2.-5. mesto na predtekmovanju: ocena +10%
- 6-10. mesto na predtekmovanju: ocena +5%
- Uporaba drugih virov podatkov, ki vam izboljša rezultat (recimo vremenskih, zgolj prazniki ne štejejo!): +10%.

**Oddaja**: Na strani tekmovanja oddajte končne napovedi, na spletni učilnici pa poročilo in izvorno kodo (kot pri preteklih domačih nalogah). Poročilo mora biti napisano s predpisano predlogo. V posameznem dnevu lahko oddate največ štiri napovedi testnih podatkov.

**Poročilo** naj vsebuje naslednja poglavja:

- **Uvod** (enostavčni opis cilja domače naloge)
- **Ocenjevanje točnosti**, kjer v enem odstavku opišete, kako ste interno, na učnih podatkih, ocenjevali točnost.
- **Napovedni modeli**, kjer opišete metode priprave podatkov, ki ste jih za oblikovanju napovedi uporabili. Vsako metodo poimenujte (akronim), njen opis pa naj ne bo daljši od 5 vrstic. Te opise postavite v latex okolju description (\item[ime] Opis.). Predvidevamo, da boste razvili nekaj metod, katerih rezultate boste preverili na strežniku. Med njimi izberite do pet najbolj zanimivih ali najbolj točnih, in te vključite v poročilo o domači nalogi.
- **Rezultati**, s tabelo z rezultati (ime metode, oddaja, ocena s preverjanjem na učnih podatkih, ocena na tekmovalnem strežniku) ter kratek, enoodstavčni komentar. V tabeli z znakom "*" označite končno oddajo.

## Napotki

### Linearna regresija

Uporabite lahko linearno regresijo, ki smo jo napisali izvajalci predmeta. Pogumnejši naj jo poskusijo implementirati sami. Nekaj priročne kode in preprostih primerov smo objavili na strani tekmovanja.

### Preverjanje na učnih podatkih

Na učnih podatkih preverjajte z enako metriko, kot jo uporabljamo na strežniku (MAE). Bodite pozorni na to, da je celotna metodologija podobna tehniki na tekmovanju. Na tekmovanju napovedujete cel neznan mesec. Poskrbite, da boste tudi vaše učne podatke zato razdelili po večjih časovnih enotah.

### Matrike v Pythonu

Za delo s numeričnimi podatki skoraj obvezno uporabite knjižnjico numpy, saj bo tako vaša koda delovala veliko hitrejo.

**Redke matrike**

Če so vaše matrike podatkov redke (večina elementov je enakih nič), se vam za hitrejše izvajanje splača uporabiti redke matrike iz knjižnjice scipy.sparse. Seveda šele, ko zahtevne dele opravite zgolj z matričnimi operacijami. Učne podatke predstavite s tabelo tipa csr_matrix. Najhitreje jo ustvarite, če podate samo neničelne elemente. Če želimo matriko velikosti 3x4 napolniti z enicami na vseh kotih (ostalo naj bodo ničle), lahko zapišemo:

```
X = scipy.sparse.csr_matrix(([1,1,1,1], [ [0,0,2,2], [0,3,0,3] ]))
```

Kadarkoli boste nato delali s podatkovno matriko pazite, da namesto operacij knjižnjice numpy uporabite operacije iz scipy.sparse. Na primer, namesto `numpy.hstack` uporabite `scipy.sparse.hstack`. Pri matričnih operacijah pazite, da bo prvi operand vedno redka matrika. Pri množenju matrik lahko vrstni red operandov obrnete, če upoštevate $(AB)^T = B^T A^T$.

◄ 2. domača naloga: podobnost jezikov

Jump to...

4. domača naloga: logistična regresija ►