Önálló kutatási feladat 2020

Mérföldkő

A projekt minden adata és kódja a GitHub repository-ban érhető el.

Adatok előfeldolgozása

Az adatok előfeldolgozását a két adathalmazunkra bontva fogjuk ismertetni, először a tőzsdei adatokat, majd a twitter adatokat.

Tőzsdei adathalmaz

Az előfeldolgozás itt nem volt nagyon komplex feladat, a részvény nyitó és záró értékéből minden napra kiszámoltuk az aznapi változását, ami így előjelesen jelent meg a táblázatunkban. Ezután egy függvénnyel felcímkéztük az egyes rekordokat, és ebben a függvényben állítható az "érzékenység", azaz, hogy mekkora változástól regisztráljuk a táblában, mint valódi változás. Ezt dollárban tudjuk állítani, ki fogjuk próbálni különböző érzékenységekre is a modellünket. A címke algoritmusában adunk nulla címkéket is, ezt egyelőre nem tervezzük használni az osztályozásban, viszont a jövőben ez megváltozhat, szóval most benne hagyjuk.

Twitter adatok

Itt első körben a dátumozással és a redundáns attribútumokkal küzdöttünk meg, mindkét halmazban egységesre állítottuk a dátumokat, majd kiszűrtük azokat a rekordokat amik kívül esnek a tényleges elnöki perióduson. Itt nagy segítségünkre volt a *pandas* csomag, ami tök kényelmesen kezel nagyon furcsa formátumú dátumokat is. Az adathalmazból minden attribútumot kidobtunk, ami nem a dátum vagy a szöveg, hiszen a benne lévő képek és a különböző statisztikák teljesen irrelevánsak a modellezésünk szempontjából. A nagyobb kihívást a szövegben szerplő linkek és tagek jelentették. Ezekkel kapcsolatban arra jutottunk, hogy jobb lesz kiszűrni őket teljesen, mert az NLP algoritmusunk a jövőben azt hihetné hogy ezek önmagukban is jelentéssel bírnak, ami ronthatná a pontosságot, de természetesen észben tartjuk ez a döntést, amikor a potenciális hibaforrásokat elemezzük majd a végén.

A megoldásunk ezekre regex segítségével született:

```
def tags(row): #Twitter tag-elést szűrő fv
    return re.sub(r'@ ?(\w){1,15}', " ", row['text'] )

def urls(row): # linkeket szűrő fv
    return re.sub(r'(https?:\/\/)?(www\.)?[-a-zA-Z0-9@:%._\+~#=]{1,256}\.[a-zA-Z0-
9()]{1,6}\b([-a-zA-Z0-9()@:%_\+.~#?&\/=]*)', " ", row['text'] )
```

Ezek vegyesen állnak fórumokon talált és általunk írt elemekből, az alap adataink helyenként furcsa formázása miatt kellett benne módosításokat eszközölnünk.

Az utolsó lépés a táblák illesztése volt, egyszerűen dátum szerint csináltuk. Az egy napra eső tweetek esetében úgy döntöttünk, kipróbáljuk úgy is, hogy összefűzzük az egy napi tweeteket, és úgy is hogy külön rekordok azonos címkével.

Az adatokat kiexportáltuk mindkét verzióban, a cdj a "concated joined data"-t jelenti, az az ahol a tweetek egy napra össze vannak fűzve, ez a file a tisztított_adatok.xlsx, a másik a non_cdj.xlsx.

Baseline modellek

Az első modellek felállításánál a scikit learn CountVectorizer() csomagját használtuk ahhoz, hogy a modellek számára is fogyasztható attribútumvektorokat állítsunk elő. Ez az első Bag of Words modell még nem egy világmegváltó dolog, de kiindulásnak jó.
A két modell a logisztikus regresszió és a random forest lett, mindkettő az sklearn csomagból jött. Ezeket kipróbálva nem érünk el túl jó eredményeket, olyan 50-55 százalékos Accuracy mellett predictelnek, szóval elég könnyű lesz majd felülmúlni őket.

Hogyan tovább?

A továbbiakban elkezdünk a neurális hálózatokkal játszani, itt a Keras és vele a Tensorflow az egyik esélyes, valamint ligába száll még a Pytorch is.
(*Ehhez kapcsolódóan kaphatunk valami tippet, hogy melyikben lenne érdemes elmélyülni?*)

Hivatkozások

Itt szeretnék listázni pár cikket-videót-kurzust, ami segítségünkre volt (van) a feladatban.

- A Stanford NLP kurzusa, ami Youtube-on elérhető
- Egy Real Python cikk, ami sokszor használt kapaszkodóul.