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, ammi 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 benne hagyjuk egyelőre.

```
def label(row, threshold):
    if abs(row['Change']) > threshold:
        if row['Change'] > 0: return int(1)
        else: return int(-1)
    else: return int(0)

def generate_data(sensitivity):
    stock['Label'] = stock.apply (lambda row: label(row, sensitivity), axis=1)
    stock_ready = stock.drop(columns=['Open', 'Close', 'Change'])
    stock_ready = stock_ready[stock_ready.Label != 0]
    return stock_ready
```

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.

A megoldásunk ezekre regex segítségével született:

```
def handlehandler(row):
    return re.sub(r'@ (\w){1,15}', " ", row['Tweet-text'] )

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

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

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