Sentimentalna analiza i predviđanje kretanja cijena dionica koristeći Twitter podatke

Džemil Džigal
Midhat Hodo
Ena Pirija

Dionice, indeksi, Calls & Puts

Predviđanje cijena pojedinih dionica je izuzetno resursno intenzivno i neisplativo

Predviđanje cijena indeksa je lakše jer isti objedinjuju više dionica u istu grupu

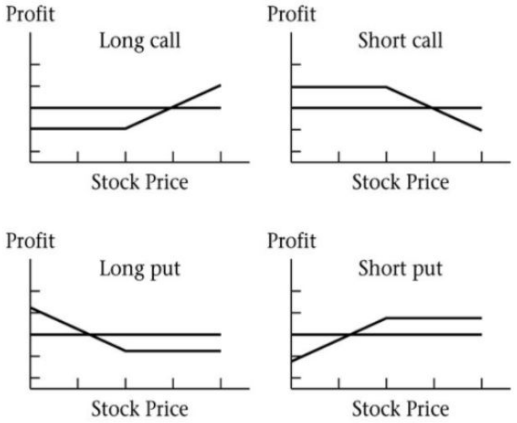
Calls - “klađenje” da će cijena dionice porasti

Put - “klađenje” da će cijena dionice pasti

Trajanje Call-ova i Put-ova. Kad zakazati strike-out? Long vs Short-term strike.

Long - “siguran ali mal profit”

Short - “rizičan ali visok profit”



Uticaj socijalnih mreža na cijene dionica

Na cijene dionica utiču ne samo standardni faktori kao stanje u industriji, potražnja i slično.

Utjecajni ljudi sa uspostavljenim dosegom na socijalnim mrežama utiču na cijene.

Elon Musk

Bitcoin Crash

Pad berze 2008 je plod rada ljudi, ne kolapsa industrije (hedge funds).

Sentimentalna analiza Tweet podataka

Primijećen je uticaj tweet-ova na cijene dionica na berzi.

Ovi podaci se mogu analizirati kako bi se odredio sentiment kretanja cijene dionica.

Određene riječi u tweet-ovima su često bile povezane sa rastom ili padom cijena dionica.

Crash, bearish, sell, stop, short su često povezane sa negativnim sentimentom i padom dionica.

Boom, bullish, buy, start, good su često povezane sa pozitivnim sentimentom i rastom cijena.

Earning, trading, market, new su često povezane sa neutralnim sentimentom i održavanjem cijena.

Sentimentalna analiza

Koristeći labele i vrijednosti u dataset-u, možemo vizualizirati sentiment i analizirati isti

```
def wordcloud_by_sentiment(sentiment):
    not_ticker = []

    for text in data[data['sentiment']==sentiment]['text2']:
        for word in text.split():
            if word.upper() not in ticker_dic:
                not_ticker.append(word.lower())

    words = ' '.join([word for word in not_ticker])
    wordcloud = WordCloud(
        width=800, height=400, background_color='white', max_font_size=110, max_words=100).\
        generate(words)
    plt.figure(figsize=(14, 7))
    plt.imshow(wordcloud, interpolation="bilinear")
    plt.title('Words in '+ sentiment.capitalize() + ' tweets\n', fontsize=32)
    plt.axis('off')
    plt.show()
```

Riječi sa pozitivnim sentimentom

Riječi sa neutralnim sentimentom

Riječi sa negativnim sentimentom

Priprema podataka

Klasična priprema tekstualnih podataka za NLP.

Uklanjanje UNICODE i specijalnih karaktera.

Koristiti emoji-e za sentiment ako sentiment nedostaje u podacima (NaN).

```
def arrange_text(ds):
    ds['text2'] = ds['text'].apply(emoji)
    ds['text2'] = ds['text2'].apply(handle)
    ds['text2'] = ds['text2'].apply(specialcode)
    ds['text2'] = ds['text2'].apply(hashtag)
    ds['text2'] = ds['text2'].apply(url)
    ds['text2'] = ds['text2'].apply(pic)
    ds['text2'] = ds['text2'].apply(html_tag)
    ds['text2'] = ds['text2'].apply(onlychar)
    ds['text2'] = ds['text2'].apply(decontracted)
    ds['text2'] = ds['text2'].apply(onlychar)
    ds['text2'] = ds['text2'].apply(tokenize_stem)
    ds['text2'] = ds['text2'].apply(remove_stopwords)
```

Metode predikcije sentimenta

Logistička regresija - Jednostavan metod, brz i adekvatan za podatke koje koristimo.

Naive Bayes - Brz metod, jednostavan ali potencijalni problem uvode novi podaci (uvode ovisnost).

SVM - Robustan i brz metod, više otporan na probleme prethodnih metoda. Veća tačnost.

Logistička regresija

Sklean paket, LogisticRegression biblioteka

```
#logistic regression
from sklearn.linear_model import LogisticRegression

classifier = LogisticRegression()
classifier.fit(X_train, y_train)
score = classifier.score(X_test, y_test)


print("Accuracy:", score)
```

Tačnost, 77.76%.

Naive Bayes

Sklean paket, naive_bayes modul, BernoulliNB klasa

```
1  #Naive Bayes
2  from sklearn.naive_bayes import GaussianNB
3  from sklearn.naive_bayes import CategoricalNB
4  from sklearn.naive_bayes import BernoulliNB
5
6  clf = BernoulliNB()
7  clf.fit(X_train, y_train)
8  score_clf = clf.score(X_test, y_test)
9  print(score_clf)
```

Tačnost, 77.76%.

SVM

Sklean paket, svm modul, SVC klasa

```
#SVM
import numpy as np
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
clf_svm = make_pipeline(StandardScaler(with_mean=False), SVC(gamma='auto'))
clf_svm.fit(X_train, y_train)
clf_svm.score(X_test, y_test)
```

Tačnost, 81.44%.

Dalji rad i zaključak

RNN ? Da li vrijedi vremena treniranja za poboljšanje tačnosti od oko 5-10% (ako i toliko)? Vrijede li većeg vremena inferiranja?

Ako se implementira distribuirani sistem, performanse postaju vrlo važne jer se planira sugerisati kupovina **short-term** call-ova i put-ova na SNP500 indeks.

Umjesto RNN-ova, Fuzzy kontrolni sistemi? Manja tačnost, manje podataka ali veća brzina?

Koliko brzo je prebrzo?

Šta su prihvatljive performanse i rezultati?

Kod i dataset dostupni na:

dzemildzigal/DAP (github.com)

Hvala na pažnji!

Q&A