Dwi Aji Kurniawan
15/330174/PTK/39360

Analisis Ulasan Film

Latar belakang

Pada setiap produk, pastilah pengguna memiliki pendapat mengenai produk tersebut. Pendapat tersebut dapat berupa pendapat positif dan negatif. Dengan berkembangnya internet, pendapat itu dapat dibagikan kepada pengguna atau calon pengguna yang lain memberikan pandangan lain terhadap produk tersebut. Salah satu contoh dari produk tersebut adalah film. Pada saat ini sudah sangat banyak situs web yang menyediakan fasilitas untuk berbagi pendapat mengenai berbagai film.

Pengguna dapat memanfaatkan situs-situs web tersebut untuk memberikan komentar terhadap berbagai film. Komentar tersebut dapat bernilai positif maupun negatif. Nilai positif dan negatif ini disebut sentimen. Dengan memanfaatkan data teks komentar pengguna sekaligus penilaian sentimen positif maupun negatif, kita dapat menentukan hal-hal yang dianggap penting oleh pemberi komentar baik positif maupun negatif. Dari hal tersebut, dapat juga dilihat hal-hal yang menjadi alasan pengguna memberikan komentar positif maupun negatif.

Hasil analisis dari data tersebut dapat dimanfaatkan oleh para kreator film untuk mengetahui aspek-aspek yang diperhatikan oleh penonton ketika menonton film sehingga ke depannya kreator film dapat membuat film yang lebih dapat memuaskan penontonnya.

Data yang Digunakan

Data yang digunakan dalam analisis ini didapatkan dari alamat web http://www.cs.cornell.edu/People/pabo/movie-review-data/. Data set yang tersedia tersebut berisi 1000 data komentar positif dan 1000 data komentar negatif pada situs web ulasan film. Data tersebut sudah mengalami sedikit preprocessing dengan tokenisasi dan penyeragaman semua huruf menjadi *lowercase*. Data tersebut juga cenderung sudah bebas dari singkatan, kecuali singkatan umum seperti *it's*, *doesn't*, dan *don't*.

Alat yang Digunakan

Analisis ini menggunakan alat-alat sebagai berikut:

- Notebook ASUS A455LD produksi tahun 2014 dengan prosesor Intel(R) Core(TM) i5-4210U 1.70-2.40 GHz, RAM 8 GB,
- Sistem operasi Ubuntu 16.04.1 LTS pada notebook,
- Intepreter Python versi 2.7.12,
- Pustaka NLTK (*Natural Language Toolkit*) versi 3.2.1 beserta corpus-corpus yang tersedia pada pustaka tersebut seperti wordnet dan *stopwords*.

Metode

Analisis ini menggunakan alat berupa bahasa Python, dengan versi 2.7.12. Analisis ini juga menggunakan pustaka NLTK yang terdapat pada bahasa Python sebagai alat analisis teks yang cukup lengkap. Kode Python ditulis dan dieksekusi pada *notebook* untuk memproses data set ulasan film yang telah didapat sebelumnya.

Secara garis besar, proses analisis ulasan film yang dilakukan adalah sebagai berikut:

- **Menghilangkan tanda baca**

Analisis ini hanya melakukan analisis terhadap kata-kata yang terdapat pada dataset, sehingga tanda baca seperti titik (.), koma (,), dan semacamnya tidak diperlukan. Tanda-tanda baca tersebut dihilangkan dari teks dengan menggunakan *regular expression*. Secara teknis, kodenya adalah "`re.sub(r'\s[^a-zA-Z0-9]',' ', teks)`".

- **Tokenisasi**

Tokenisasi dilakukan untuk mengubah teks yang bertipe data *string* menjadi *array* yang berisi token-token. Proses tokenisasi pada analisis ini menggunakan metode unigram, yang menganggap satu kata sebagai satu token.

- ***Stemming***

Proses *stemming* merupakan proses pemotongan suatu kata sehingga kata-kata yang memiliki kata dasar yang sama mempunyai bentuk yang sama. Contohnya adalah kata "*study*", "*studying*", "*studied*", "*studies*" jika dilakukan *stemming* dengan algoritma Porter (Porter, 1980) akan menjadi bentuk yang sama, yaitu "*studi*". Hal ini akan mempermudah proses analisis teks karena pada dasarnya kata-kata tersebut mempunyai makna "belajar".

Proses *lemmatization*, merupakan pengembangan dari metode *stemming*. Proses *lemmatization* akan mengubah semua kata menjadi kata dasarnya. Perbedaan yang paling terlihat dari dua metode tersebut adalah, *stemming* dapat dilakukan dengan metode berbasis *rule*, sedangkan *lemmatization* memerlukan data set sebagai acuan untuk melakukan *lemmatization*. Untuk hasilnya, *lemmatization* tidak mengubah kata "studying" dan "studying" menjadi "study" maupun "studi", akan tetapi tetap menjadi kata asalnya. Oleh karena itu, analisis ini menggunakan *stemming* agar kata -ing dan -ed menjadi bentuk yang sama.

- **Penghilangan *stopwords***

Stopwords merupakan kata-kata yang memiliki makna yang tidak terlalu signifikan pada teks. Penghilangan *stopwords* dapat meningkatkan kualitas serta menurunkan beban sistem pada saat analisis. Daftar *stopwords* yang digunakan merupakan *stopwords* bawaan dari pustaka NLTK.

- **POS Tagging dan penyaringan tag**

POS (Part of Speech) tagging adalah proses pemberian tag pada setiap kata dengan tag tipe kata tersebut. Menurut dokumentasi NLTK, secara umum terdapat 12 jenis tag POS.

Tag	Meaning	English Examples
ADJ	adjective	*new, good, high, special, big, local*
ADP	adposition	*on, of, at, with, by, into, under*
ADV	adverb	*really, already, still, early, now*
CONJ	conjunction	*and, or, but, if, while, although*
DET	determiner, article	*the, a, some, most, every, no, which*
NOUN	noun	*year, home, costs, time, Africa*
NUM	numeral	*twenty-four, fourth, 1991, 14:24*
PRT	particle	*at, on, out, over per, that, up, with*
PRON	pronoun	*he, their, her, its, my, I, us*
VERB	verb	*is, say, told, given, playing, would*
.	punctuation marks	*. , ; !*
X	other	*ersatz, esprit, dunno, gr8, univeristy*

Pada analisis ini akan dilakukan penyaringan dengan hanya mengambil tag dengan tipe NOUN atau biasa disingkat dengan NN. Hal ini dilakukan karena tujuan analisis ini adalah untuk mengetahui hal-hal yang dianggap penting oleh pengguna sebagai aspek penilaian sebuah film. Hal-hal tersebut pastilah berupa kata benda.

- **Penghitungan kemunculan kata pada dataset**

Proses ini akan menghitung banyaknya kemunculan suatu kata benda pada dataset komentar positif dan dataset komentar negatif. Dengan mengetahui banyaknya kemunculan kata benda pada dataset, akan diketahui pula hal-hal apa saja yang menjadi topik diskusi pada dataset tersebut. Kata-kata yang paling banyak muncul pada dataset menjadi topik yang paling sering disebut pada dataset.

- **Penghitungan *document frequency* untuk setiap kata**

Document frequency secara sederhananya adalah banyaknya dokumen yang memuat suatu kata. Hal ini sangat berguna untuk mengenali aspek-aspek apa saja yang disinggung oleh lebih banyak orang. Tentunya aspek tersebut perlu diperhatikna untuk pembuatan film berikutnya.

Hasil

Kode python dieksekusi pada notebook dan memberikan ringkasan hasil pada terminal serta memberikan output berupa beberapa file CSV (Comma Separated Value). Kode dan hasil analisis dapat diakses secara online di https://github.com/dwiajik/tugas-bi. Berikut ini adalah ringkasan hasil analisis:

- **Kemunculan kata**

Positive Dataset	
Word	**Freq**
film	5817
movi	2681
charact	1767
time	1523
scene	1332
stori	1042
way	1024
life	984
perform	857
year	826
thing	801
end	765
work	721
man	702
world	641
role	635
star	620

Negative Dataset	
Word	**Freq**
film	4702
movi	3240
charact	1567
time	1376
scene	1272
plot	909
thing	845
way	831
stori	763
end	688
action	648
director	612
year	606
actor	602
man	580
work	563
look	536

Hasil tersebut adalah contoh hasil dari 20 kata yang paling sering muncul di dataset komentar positif maupun negatif. Kolom sebelah kiri merupakan kata hasil *stemming*, lalu kolom di sebelah kanan merupakan frekuensi kemunculan kata tersebut pada dataset. Terlihat bahwa lima kata yang sering muncul pada dataset positif maupun negative adalah identik, kemudian hasilnya baru bervariasi. Dari ringkasan tersebut terlihat bahwa aktor dan aktris merupakan aspek yang sangat penting pada film, diikuti dengan waktu, *scene*, dan plot cerita. Bahkan di dataset negatif juga terdapat kata *director* yang berarti sutradara.

- ***Document frequency***

Positive DF	
Word	**DF**
film	903
movi	770
time	722
charact	691
scene	594
way	559
stori	506
year	500
perform	489
life	471
thing	471
end	459
work	457
man	413
director	405
play	390
role	387

Negative DF	
Word	**DF**
film	875
movi	812
time	672
charact	669
scene	566
plot	523
way	514
thing	483
stori	451
director	446
end	434
year	393
actor	393
work	392
play	374
interest	363
look	361

Hasil tersebut adalah contoh hasil dari 20 kata yang muncul di paling banyak dokumen. beserta. Kolom sebelah kiri merupakan kata hasil *stemming*, lalu kolom di sebelah kanan merupakan banyaknya dokumen yang memuat kata tersebut. Terlihat bahwa lima kata yang sering muncul pada dataset positif maupun negative adalah identik, kemudian hasilnya baru bervariasi. Hasilnya cenderung sama dengan hasil sebelumnya yang berarti kedua hasil tersebut adalah berhubungan.

Lampiran

```
from os import listdir
from os.path import isfile, join
import operator
import re

import nltk
from nltk.corpus import stopwords
from nltk.stem.snowball import SnowballStemmer

english_stopwords = set(stopwords.words('english'))
stemmer = SnowballStemmer("english")

pos_path = 'Movie Reviews Data Set/review_polarity/txt_sentoken/pos'
neg_path = 'Movie Reviews Data Set/review_polarity/txt_sentoken/neg'

pos_files = [join(pos_path, f) for f in listdir(pos_path) if isfile(join(pos_path, f))]
neg_files = [join(neg_path, f) for f in listdir(neg_path) if isfile(join(neg_path, f))]

pos_words = {}
neg_words = {}

pos_df = {}
neg_df = {}

for file_path in pos_files:
        with open(file_path) as f:
        df = {}
        for line in f:
                line = re.sub(r'\s[^a-zA-Z0-9]',' ', line)
                tokens = line.split()
                tokens = [stemmer.stem(token).encode("ascii") \
                for token in tokens \
                if '\'s' not in token \
                and '\'t' not in token]
                tokens = nltk.pos_tag(tokens)
                for token in tokens:
                if 'NN' in token[1] and token[0] not in english_stopwords:
                        try:
                        df[token] += 1
                        except:
                        df[token] = 1
                        try:
                        pos_words[token] += 1
                        except:
                        pos_words[token] = 1
        for key in df:
                try:
                pos_df[key] += 1
                except:
                pos_df[key] = 1


for file_path in neg_files:
        with open(file_path) as f:
        df = {}
        for line in f:
                line = re.sub(r'\s[^a-zA-Z0-9]',' ', line)
                tokens = line.split()
                tokens = [stemmer.stem(token).encode("ascii") \
```

```
                for token in tokens \
                if '\'s' not in token \
                and '\'t' not in token]
                tokens = nltk.pos_tag(tokens)
                for token in tokens:
                if 'NN' in token[1] and token[0] not in english_stopwords:
                        try:
                        df[token] += 1
                        except:
                        df[token] = 1
                        try:
                        neg_words[token] += 1
                        except:
                        neg_words[token] = 1
        for key in df:
                try:
                neg_df[key] += 1
                except:
                neg_df[key] = 1

pos_words = sorted(pos_words.items(), key=operator.itemgetter(1), reverse=True)
neg_words = sorted(neg_words.items(), key=operator.itemgetter(1), reverse=True)

pos_df = sorted(pos_df.items(), key=operator.itemgetter(1), reverse=True)
neg_df = sorted(neg_df.items(), key=operator.itemgetter(1), reverse=True)

with open('pos.csv', 'a') as f:
        for item in pos_words:
        f.write('{0},{1},{2}\n'.format(item[0][0], item[0][1], item[1]))

with open('neg.csv', 'a') as f:
        for item in neg_words:
        f.write('{0},{1},{2}\n'.format(item[0][0], item[0][1], item[1]))

print 'Total pos words:', len(pos_words)
for word in pos_words[:40]:
        print word

print 'Total neg words:', len(neg_words)
for word in neg_words[:40]:
        print word

with open('pos_df.csv', 'a') as f:
        for item in pos_df:
        f.write('{0},{1},{2}\n'.format(item[0][0], item[0][1], item[1]))

with open('neg_df.csv', 'a') as f:
        for item in neg_df:
        f.write('{0},{1},{2}\n'.format(item[0][0], item[0][1], item[1]))

print 'IDF pos'
for word in pos_df[:40]:
        print word

print 'IDF neg'
for word in neg_df[:40]:
        print word
```