Data Mining Review Hotel The Apurva Kempinski Bali Menggunakan BeautifulSoup

Dibuat Oleh: Stefannus Christian

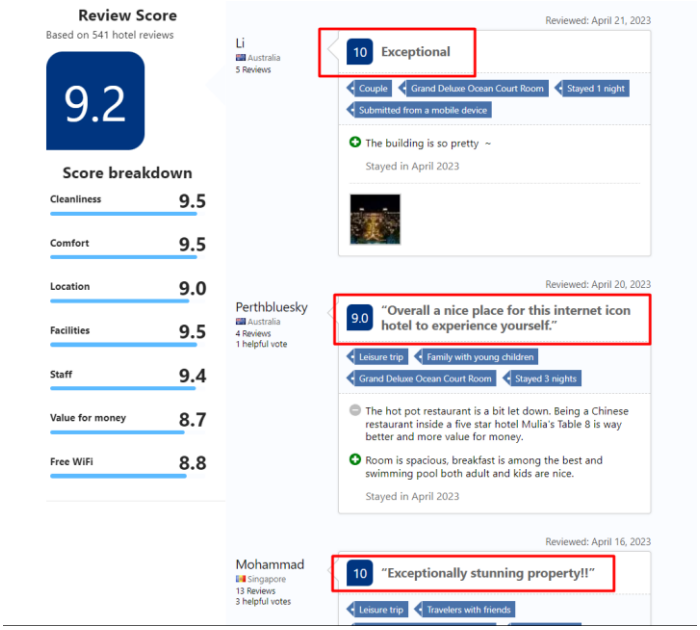
Link Github Project: https://github.com/StefannusChristian/Data-Mining/tree/main/apurva-kempinski-bali-web-scraping

Berikut merupakan link – link referensi belajar yang saya gunakan ketika mengerjakan projek ini:

1. Link Belajar Beautiful Soup: https://www.youtube.com/watch?v=XVv6mJpFOb0
2. Link Belajar WordCloud: https://www.youtube.com/watch?v=HcKUU5nNmrs

Disini saya menganalisis review penjunjung hotel yang termuat dalam website booking.com dengan link berikut ini: https://www.booking.com/reviews/id/hotel/the-apurva-kempinski-bali.html?sid=d5b706fb1e392d7bdb21e2beb7949c62&aid=356980&order=featuredreviews&label=gog235jc-1FCA0oaEIZdGhlLWFwdXJ2YS1rZW1waW5za2ktYmFsaUgzWANoalgBAZgBMbgBF8gBDNgBAegBAfgBAogCAagCA7gCnNKLogbAAgHSAiRhNTZlZDcxNS0yNTIzLTRmY2QtOTIwMy05MmRhYTVlYzdkOWLYAgXgAgE&hp_nav=0&page=1&customer_type=total&rows=75&old_page=0&r_lang=en

Review – review tulisan di kotak merah pada gambar dibawah ini adalah review yang akan saya analisis.



Langkah – Langkah:

1. Install Library Python yang dibutuhkan

Jalankan ketiga perintah dibawah ini pada untuk menginstall library beautifulsoup yang akan digunakan untuk scraping, menginstall library wordcloud yang akan digunakan

untuk membuat wordcloud dari hasil scraping, dan menginstall library lxml yang dibutuhkan untuk memproses html.

- *pip install beautifulsoup4*
- *pip install wordcloud*
- *pip install lxml*

2. Import Library yang diperlukan

```
from requests import get
from re import sub
from bs4 import BeautifulSoup
from wordcloud import WordCloud, STOPWORDS
from time import perf_counter
```

- Library requests method get digunakan untuk mengambil text dari situs booking.com.
- Library re (regex) method sub digunakan untuk membersihkan text hasil dari scraping seperti membuang spasi berlebih pada text hasil scraping.
- Library BeautifulSoup akan digunakan untuk web scraping.
- Library WordCloud akan digunakan untuk membuat wordcloud.
- Library time method perf_counter akan digunakan untuk menghitung waktu eksekusi program seperti waktu eksekusi scraping per page dan juga waktu eksekusi membuat word cloud.

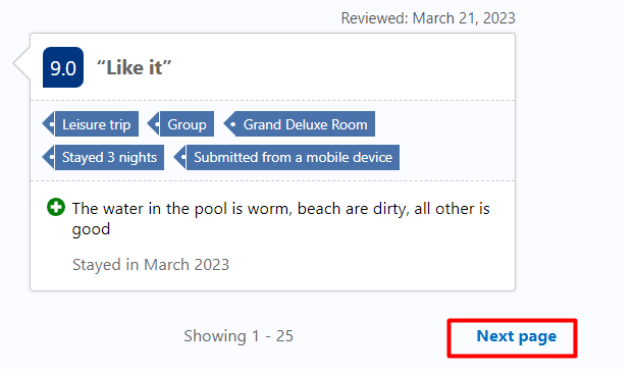
3. Scraping Web Booking.com

```
def scrape_reviews():
    scrape_start_time = perf_counter()
    all_reviews = []
    for page in range(1,9):
        page_start_time = perf_counter()
        print(f"Scraping page {page} ...")
        url = f"https://www.booking.com/reviews/id/hotel/the-apurva-kempinski-bali.html?aid=356980&
        label=gog235jc-1DCA0oaEIZdGhlLWFwdXJ2YS1rZW1waW5za2ktYmFsaUgzWANoaIgBAZgBMbgBF8gBDNgBA-gBAfgBAogCAagCA7qCnNKLogbAAgHSAiRhNTZlZDcxNS0yNTIzL
        TRmY2QtOTIwMy05MmRhYTVlYzdkOWLYAgTgAgE&sid=d5b706fb1e392d7bdb21e2beb7949c62&customer_type=total&hp_nav=0&old_page=0&order=featuredreviews&
        page={page}&r_lang=en&rows=75&"

        html_text = get(url).text
        soup = BeautifulSoup(html_text, 'lxml')
        reviews = soup.find_all('div',class_="review_item_header_content")

        for word in reviews:
            try:
                review = word.span.text.strip()
                review = remove_special_characters(review)
                review = correct_typo(review)
                all_reviews.append(review.strip().lower())
            except: pass

        page_end_time = perf_counter()

        print(f"Scraping page {page} finished!")
        print(f"Scraping page {page} took {page_end_time-page_start_time:.4f} seconds\n")

    scrape_end_time = perf_counter()
    print(f"Scraping took {scrape_end_time-scrape_start_time:.4f} seconds!\n")

    return all_reviews
```

Disini saya membuat function scrape_reviews() untuk scraping web booking.com untuk hotel Apurva Kempinski Bali. Saya akan menjelaskan kode saya line per line dengan memberikan terlebih dahulu cuplikan kode kemudian menjelaskan apa yang kode tersebut lakukan.

```
scrape_start_time = perf_counter()
all_reviews = []
```

Variabel scrape_start_time akan mencatat waktu Ketika function scrape_review dijalankan. Variabel ini nantinya akan digunakan untuk menghitung waktu yang digunakan untuk menjalankan scraping. Variabel all_reviews digunakan untuk menyimpan seluruh review – review hasil scraping.



Terdapat 8 page review pada website ini. Website berikutnya dapat diakses dengan menekan next page seperti gambar diatas. Tetapi disini, saya menemukan bahwa URL yang digunakan untuk page berikutnya dapat diubah dari bagian url. Dapat dilihat dari gambar dibawah ini bahwa saya tinggal mengganti url web pada bagian {page} dan reviewnya pun akan berubah ke review halaman {page}.

```
url = f"https://www.booking.com/reviews/id/hotel/the-apurva-kempinski-bali.html?aid=356980&
label=gog235jc-1DCA0oaEIZdGhlLWFwdXJ2YS1rZW1waW5za2ktYmFsaUgzWANoaIgBAZgBMbgBF8gBDNgBA-gBAfgBAogCAagCA
7gCnNKLogbAAgHSAiRhNTZlZDcxNS0yNTIzLTRmY2QtOTIwMy05MmRhYTVlYzdkOWLYAgTgAgE&
sid=d5b706fb1e392d7bdb21e2beb7949c62&customer_type=total&hp_nav=0&old_page=0&order=featuredreviews&
page={page}&r_lang=en&rows=75&"
```

```
for page in range(1,9):
    page_start_time = perf_counter()
    print(f"Scraping page {page} ...")
    url = f"https://www.booking.com/reviews/id/hotel/the-apurva-kempinski-bali.html?aid=356980&
    label=gog235jc-1DCA0oaEIZdGhlLWFwdXJ2YS1rZW1waW5za2ktYmFsaUgzWANoaIgBAZgBMbgBF8gBDNgBA-gBAfgBAogCAag
    7gCnNKLogbAAgHSAiRhNTZlZDcxNS0yNTIzLTRmY2QtOTIwMy05MmRhYTVlYzdkOWLYAgTgAgE&
    sid=d5b706fb1e392d7bdb21e2beb7949c62&customer_type=total&hp_nav=0&old_page=0&order=featuredreviews&
    page={page}&r_lang=en&rows=75&"
```

Maka dari itu karena adanya 8 page pada website ini, saya melakukan for loop untuk mengiterasi dari page 1 – 8. Variabel page_start_time digunakan untuk menghitung start waktu dari ketika halaman {page} mulai di scrape dan nantinya akan digunakan untuk menghitung waktu yang dibutuhkan untuk scraping halaman {page}.

```
html_text = get(url).text
soup = BeautifulSoup(html_text, 'lxml')
```

Disini variable html_text digunakan untuk mendapatkan struktur html dari url yang diberikan. Variabel soup digunakan untuk menginisialisasi BeautifulSoup yang akan

digunakan untuk scraping. Saya tidak akan menjelaskan secara detail apa yang dilakukan kedua variabel diatas, penjelasan lebih lanjut dapat ditonton disini.

Cara kerja beautifulsoup (bs) adalah bs dengan bantuan lxml (parser) akan mencari tag atau class dari elemen yang ingin di scrape. Disini, menggunakan dev tools browser untuk inspect element yang ingin dicari apa nama classnya dan class tersebut berada pada tag apa.

```
<div class="review_item_header_content
  high_score_word
  review_item_header_scoreword
"> == $0
  <span itemprop="name"> Exceptional </span>
</div>
```

Dapat dilihat bahwa kata review yang ingin saya ambil yaitu Exceptional berada pada di tag span dengan parent tag div dengan class review_item_header_content. Maka cara saya mendapatkan seluruh review adalah dengan mencari seluruh div tag dengan class review_item_header_content. Berikut kode untuk melakukan hal ini.

```
reviews = soup.find_all('div',class_="review_item_header_content")
```

Tetapi disini variabel reviews belum secara spesifik mendapatkan kata review yang diinginkan. Reviews disini bentuknya adalah list yang menyimpan seluruh div dengan class review_item_header_content.

```
for word in reviews:
    try:
        review = word.span.text.strip()
        review = remove_special_characters(review)
        review = correct_typo(review)
        all_reviews.append(review.strip().lower())
    except: pass
```

Berikut cara untuk mendapatkan kata review yang diinginkan. Iterasilah list reviews dan berdarkan struktur kode html yang telah saya berikan diatas, review word nya terdapat pada span sehingga kita perlu mengakses span tersebut dengan cara word.span. Kemudian .text digunakan untuk mendapatkan kata dari span tersebut dan .strip() digunakan untuk menghilangkan spasi berlebih dari string tersebut. Kemudian review tersebut akan dibersihkan dan di append ke list all_reviews. Saya menggunakan function lower() untuk membuat review menjadi huruf kecil.

```python
def remove_special_characters(review: str):
    review = review.replace("😊","")
    review = review.replace("👍","")
    review = review.replace("’","")
    review = review.replace("!","")
    review = review.replace(".","")
    review = review.replace(")","")
    review = review.replace(",","")
    review = review.replace(":","")
    review = review.replace("-","")
    review = review.replace("'","")
    return review
```

```python
def correct_typo(review: str):
    review = review.replace("pleasent","pleasant")
    review = review.replace("recomended","recommended")
    review = review.replace("recommend","recommended")
    review = review.replace("exceptionally","exceptional")
    return review
```

Kedua function diatas saya apply untuk membersihkan review sebelum dimasukkan kedalam list all_reviews. Saya replace tanda tanda baca seperti titik,koma, dll. Saya juga membenarkan typo yang saya temukan di review tersebut seperti pleasant yang seharusnya pleasant.

```python
        page_end_time = perf_counter()

        print(f"Scraping page {page} finished!")
        print(f"Scraping page {page} took {page_end_time-page_start_time:.4f} seconds\n")

    scrape_end_time = perf_counter()
    print(f"Scraping took {scrape_end_time-scrape_start_time:.4f} seconds!\n")

    return all_reviews
```

Kemudian variabel page_end_time digunakan sebagai tanda bahwa scraping page tersebut sudah selesai. Kemudian cara untuk menghitung berapa waktu yang digunakan untuk scraping satu page adalah dengan mengurangi waktu akhir dengan waktu awal. Begitu juga cara untuk menghitung waktu total yang digunakan untuk scraping seluruh page. Function ini kemudian akan return all_reviews yang isinya adalah review words yang sudah di bersihkan.

4. Menyimpan hasil scraping ke file txt

```python
def clean_reviews(reviews: list): return list(filter(None, reviews))

def convert_review_to_paragraph(reviews: list):
    review_text = " ".join(reviews)
    review_text = sub("  "," ",review_text)
    return review_text

def save_review_to_txt_file(path:str,filename:str):
    all_reviews = clean_reviews(scrape_reviews())
    print("Scraping Apurva Kempinski Bali Reviews From Booking.com Website Finished!")
    review_text = convert_review_to_paragraph(all_reviews)
    with open(path+filename, "w") as text_file: text_file.write(review_text)
```

Untuk menconvert list all_reviews menjadi word cloud, list all_reviews perlu dibuat menjadi file txt terlebih dahulu. Yang saya lakukan pertama adalah membersihkan lagi list all_reviews dengan menghilangkan NaN values, kemudian saya menggunakan method join untuk menggabungkan isi dari list tersebut dan membuatnya menjadi satu string Panjang (paragraph). Kemudian saya menggunakan method regex yaitu sub untuk menghilangkan spasi spasi dobel yang

ada pada paragraph tersebut. Setelah itu saya convert paragraph tersebut ke file txt. Hasil dari file txt nya adalah sebagai berikut.

File txt inilah yang akan digunakan untuk membuat wordcloud.

5. Membuat wordcloud

```python
def convert_review_to_wordcloud(reviews: str, wordcloud_file_name: str):
    print("Converting Reviews To Wordcloud ...")
    wordcloud_start_time = perf_counter()
    sw = list(STOPWORDS)
    stopwords = list(set(["a","overall","place","for","this","internet","icon","hotel","to","yourself","property","place","stay","the","you","should","at","least",
    "once","it","the","very","much","and","one","of","were","really","walk","need","wait","team","think","spend","bit","building","hiccups","ou","deco","dua","peak",
    "complex","decor","koral","every","towards","apurva","extremely","sure","nusa","disa","little","planet","lagoon","cant","go","take","aspect","halls","although",
    "needs","seen","returned","kempinski","ive","penny","look","trip","besides","later","already","long","back","alone","return","staying","come","will","bali",
    "everything","swimming","room","stayed","bed","truly","possibly","welcome","dark","making","improveed","real","thank","better","staff","island","whole","season",
    "going","forward","pools","rooms","check","time","despite","hotels","highly","improve","definitely"
        ]))

    sw.extend(stopwords)
    sw = list(set(sw))

    wc = WordCloud (
        background_color='white',
        stopwords=sw,
    )

    wc.generate(reviews)
    wc.to_file(wordcloud_file_name+'.png')

    wordcloud_end_time = perf_counter()
    print("\nConverting Reviews To WordCloud Finished!")
    print(f"Converting To WordCloud Takes {wordcloud_end_time - wordcloud_start_time:.4f} seconds\n")
```

Saya menggunakan library wordcloud pada python untuk membuat wordcloud. Untuk referensi nya saya belajar disini. Stopwords digunakan untuk memfilter kata – kata apa saja yang tidak diinginkan pada wordcloud. Ketika menginstall wordcloud, default STOPWORDS otomatis akan terinstall juga tetapi default STOPWORDS yang disediakan ini tidak cukup sehingga saya menambahkan stopwords berikut ini secara manual agar kata – kata ini tidak ditampilkan pada wordcloud.

```python
stopwords = list(set(["a","overall","place","for","this","internet","icon","hotel",
"to","yourself","property","place","stay","the","you","should","at","least","once",
"it","the","very","much","and","one","of","were","really","walk","need","wait",
"team","think","spend","bit","building","hiccups","ou","deco","dua","peak","complex",
"decor","koral","every","towards","apurva","extremely","sure","nusa","disa","little",
"planet","lagoon","cant","go","take","aspect","halls","although","needs","seen",
"returned","kempinski","ive","penny","look","trip","besides","later","already",
"long","back","alone","return","staying","come","will","bali","everything",
"swimming","room","stayed","bed","truly","possibly","welcome","dark","making",
"improveed","real","thank","better","staff","island","whole","season","going",
"forward","pools","rooms","check","time","despite","hotels","highly","improve",
"definitely"
    ]))
```

Output program di console ketika di run adalah sebagai berikut.

```
Scraping Apurva Kempinski Bali Reviews From booking.com Website Starting ...

Scraping page 1 ...
Scraping page 1 finished!
Scraping page 1 took 1.3910 seconds

Scraping page 2 ...
Scraping page 2 finished!
Scraping page 6 ...
Scraping page 6 finished!
Scraping page 6 took 1.1317 seconds

Scraping page 7 ...
Scraping page 7 finished!
Scraping page 7 took 1.1887 seconds

Scraping page 8 ...
Scraping page 8 finished!
Scraping page 8 took 1.5663 seconds

Scraping took 11.1786 seconds!

Scraping Apurva Kempinski Bali Reviews From Booking.com Website Finished!
Converting Reviews To Wordcloud ...

Converting Reviews To WordCloud Finished!
Converting To WordCloud Takes 0.4934 seconds

Program Takes 11.6755 seconds to execute
```

Hasil WordCloud