ANALISIS SENTIMEN DATA ULASAN PENGUNJUNG OBJEK WISATA LAWANG SEWU KOTA SEMARANG PADA SITUS TRIPADVISOR

Tugas Akhir

diajukan untuk memenuhi salah satu syarat untuk memperoleh gelar

Ahli Madya Statistika Terapan dan Komputasi

Oleh

Anggi Vivi Febrianti

4112317002

JURUSAN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS NEGERI SEMARANG

2020

ANALISIS SENTIMEN DATA ULASAN PENGUNJUNG OBJEK WISATA LAWANG SEWU KOTA SEMARANG PADA SITUS TRIPADVISOR

Tugas Akhir

diajukan untuk memenuhi salah satu syarat untuk memperoleh gelar

Ahli Madya Statistika Terapan dan Komputasi

Oleh

Anggi Vivi Febrianti

4112317002

JURUSAN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS NEGERI SEMARANG

2020

PERNYATAAN

Dengan ini, saya

nama : Anggi Vivi Febrianti

NIM : 4112317002

program studi : Statistika Terapan dan Komputasi

menyatakan bahwa tugas akhir berjudul *Analisis Sentimen Data Ulasan Pengunjung Objek Wisata Lawang Sewu Kota Semarang pada Situs TripAdvisor* ini benar-benar karya saya sendiri bukan plagiat dari karya orang lain atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku baik sebagian atau seluruhnya. Pendapat atau temuan orang atau pihak lain yang terdapat dalam tugas akhir ini telah dikutip atau dirujuk berdasarkan kode etik ilmiah. Atas pernyataan ini, saya secara pribadi siap menanggung resiko/sanksi hukum yang dijatuhkan apabila dikemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya ini.

Semarang, 29 September 2020

Anggi Vivi Febrianti

NIM 4112317002

PENGESAHAN

Tugas Akhir berjudul *Analisis Sentimen Data Ulasan Pengunjung Objek Wisata Lawang Sewu Kota Semarang pada Situs TripAdvisor* karya Anggi Vivi Febrianti NIM 4112317002 ini telah dipertahankan dalam Ujian Tugas Akhir FMIPA Universitas Negeri Semarang pada tanggal 1 Oktober 2020 dan disahkan oleh Panitia Ujian.

Semarang, 1 November 2020

Panitia

Ketua,

Dr. Sugianto, M.Si.
NIP 196102191993031001

Sekretaris,

Dr. Mulyono, M.Si.
NIP 197009021997021001

Penguji I,

Amidi, S.Si., M.Pd.
NIP 198703012014041001

Penguji II/Pembimbing,

Dr. Iqbal Kharisudin, M.Sc.
NIP 197908052005011003

MOTTO

- Belajar dari masa lalu, berusaha saat ini, sukses dimasa depan.
- Jangan pernah lupa melibatkan Allah dalam setiap urusan hidup.

PERSEMBAHAN

Untuk Ayah, Ibu dan Adik-adik
Seluruh guru dan dosen
Almamater
Sahabat dan teman
Teman-teman organisasi *Mathematics Journalism Club*
Teman-teman organisasi Gerakan Mahasiswa Anti Narkoba
Teman-teman Aurelia kos
Teman-teman Staterkom 2017

PRAKATA

Segala puji dan syukur kehadirat Allah SWT yang telah melimpahkan rahmat serta hidayah-Nya, sehingga penulis dapat menyelesaikan tugas akhir dengan judul **"Analisis Sentimen Data Ulasan Pengunjung Objek Wisata Lawang Sewu Kota Semarang pada Situs TripAdvisor"** yang disusun sebagai salah satu syarat untuk memperoleh gelar Ahli Madya pada Program Studi Statistika Terapan dan Komputasi, Fakultas Matematika dan Ilmu Pengetahuan Alam.

Menyadari bahwa terselesaikannya penulisan tugas akhir ini berkat bimbingan, arahan, bantuan dan dukungan dari berbagai pihak baik moril maupun materiel. Oleh karena itu pada kesempatan ini, penulis menyampaikan rasa hormat dan ungkapan terima kasih kepada:

1. Prof. Dr. Fathur Rokhman, M.Hum, Rektor Universitas Negeri Semarang,
2. Dr. Sugianto, M.Si., Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Semarang,
3. Dr. Mulyono, M.Si., Ketua Jurusan Matematika FMIPA Universitas Negeri Semarang,
4. Dr. Iqbal Kharisudin, M.Sc., Ketua Program Studi Statistika Terapan dan Komputasi FMIPA Universitas Negeri Semarang,
5. Dr. Iqbal Kharisudin, M.Sc., Dosen pembimbing tugas akhir, yang telah memberikan bimbingan, arahan, saran dan koreksi sehingga tugas akhir ini dapat terselesaikan,
6. Amidi, S.Si., M.Pd., Dosen penguji tugas akhir, yang telah memberikan masukan dan perbaikan pada tugas akhir ini,
7. Bapak dan Ibu dosen Jurusan Matematika yang telah memberikan ilmu yang bermanfaat kepada penulis dalam penyusunan tugas akhir ini,
8. Ayah, Ibu, kedua adikku dan keluaga yang selalu memberi motivasi, semangat dan doa dalam penyusunan tugas akhir ini,
9. Septi, Reza, Alisha, Alifia dan Linda yang sudah banyak membantu, memberi saran, memotivasi dan mendoakan dalam penyusunan tugas akhir ini,

10. Dinda, Dini dan Handika yang selalu memberikan dukungan dan doa dalam penyusunan tugas akhir ini,
11. Firsta, Dian, Desti, Elmas, Tasya, dan Listia yang selalu memotivasi dan mendoakan,
12. Teman-teman dan semua pihak lainnya, yang telah memberikan banyak bantuan dan motivasi dalam penyusunan tugas akhir.

Penulis menyadari bahwa penyusunan tugas akhir ini masih jauh dari sempurna, sehingga dengan rendah hati penulis mengharap kritik dan saran. Penulis berharap semoga tugas akhir ini dapat berguna serta bermanfaat bagi para pembaca.

Semarang, September 2020

Penulis

ABSTRAK

Febrianti, Anggi Vivi. (2020). *Analisis Sentimen Data Ulasan Pengunjung Objek Wisata Lawang Sewu Kota Semarang pada Situs TripAdvisor*. Tugas akhir, Statistika Terapan dan Komputasi Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Semarang. Pembimbing Dr. Iqbal Kharisudin, M.Sc.

Kata Kunci: Lawang Sewu, analisis sentimen, klasifikasi, *Support Vector Mechine*, *Naïve Bayes*, *Decision Tree*, *Random Forest*, *Logistic Regression*

Kemajuan pesat dibidang teknologi dan komunikasi bisa sangat dirasakan seiring dengan perkembangan internet yang semakin maju dan meluas. Bidang pariwisata tak luput dari pengaruh kemajuan teknologi. Ketersediaan data yang melimpah di internet dapat ditambang dan dianalisis untuk memperoleh informasi yang bermanfaat. Pada penelitian ini akan dianalisis sentimen pengunjung objek wisata Lawang Sewu berdasarkan ulasan pada situs TripAdvisor. Ulasan pengunjung Lawang Sewu diperoleh dengan cara *web scraping* dari situs TripAdvisor. Data ulasan berbahasa Inggris dan berbahasa Indonesia dianalisis untuk mengetahui apakah ulasan bersentimen positif atau negatif.

Tahap analisis data yaitu *text preprocessing* untuk membersihkan data, pembobotan kata, pelabelan data menjadi kelas positif dan negatif, pengklasifikasian dan visualisasi data dengan *word cloud*. Dalam penelitian ini digunakan metode klasifikasi *Support Vector Mechine, Naïve Bayes, Decision Tree, Random Forest* dan *Logistic Regression*. Nilai akurasi yang dihasilkan untuk masing-masing metode pada data ulasan berbahasa Inggris secara berturut-turut sebesar 89,3%, 87,7%, 76,3%, 87,7%, 87,7% dan untuk data ulasan berbahasa Indonesia sebesar 86,2%, 78,9%, 72,4%, 80,0% dan 80,5%. Dari kedua data ulasan didapatkan informasi bahwa mayoritas ulasan pengunjung bersentimen positif, dan metode *Support Vector Mechine* paling baik untuk mengklasifikasikan data ulasan pengunjung Lawang Sewu.

DAFTAR ISI

DAFTAR TABEL

DAFTAR BAGAN

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1. Latar Belakang

Kemajuan yang pesat dibidang teknologi dan komunikasi bisa sangat dirasakan pada saat ini, tak luput dengan perkembangan teknologi internet yang semakin maju dan meluas, kini persebaran internet sudah bisa dirasakan hingga kepenjuru negeri. Internet telah menjadi kebutuhan pokok bagi banyak orang, internet digunakan untuk mempermudah komunikasi dan menghubungkan berbagai aspek kehidupan. Berbagai kalangan umur dan jenis profesi membutuhkan dan menggunakan internet.

Internet digunakan untuk bersosialisasi secara *online* dengan media sosial, terkadang perindividu mempunyai lebih dari satu akun media sosial, diantaranya Istagram, WhatsApp, Facebook, Telegram, Twitter, dan lain sebagainya. Banyak postingan yang diunggah oleh pengguna, baik berupa foto, vidio, teks, atau bahkan hanya sebuah like dan komentar. Semua postingan yang diunggah ke media sosial adalah suatu data yang berlimpah, yang jika mampu menambang dan menganalisisnya maka akan didapatkan pengetahuan baru yang bermanfaat.

Bidang pariwisata saat ini juga sudah sangat terpengaruh oleh kemajuan teknologi dan internet, banyak dampak positif yang dirasakan dibidang pariwisata, dengan adanya kemajuan teknologi dan internet, sekarang banyak situs *online* yang menawarkan jasa *booking* hotel, restoran, penginapan, *booking* tiket transportasi seperti kereta api, pesawat, dan lainnya. Bahkan untuk orang yang sekedar ingin mencari referensi untuk rencana wisatanya, kini sudah dapat memanfaatkan banyak media sosial, untuk mencari foto-foto hingga ulasan atau *review* tentang pariwisata tersebut dari para pengunjung sebelumnya. Media sosial yang sering digunakan untuk acuan pariwisata adalah TripAdvisor, TripAdvisor adalah situs wisata *online* terbesar di dunia, situs ini dapat membantu para wisatawan baik domestik atau mancanegara dalam perencanaan perjalanan wisata mereka.

Saat ini berwisata atau *travelling* bagi sebagian orang sudah termasuk dalam kebutuhan hidup yang harus dipenuhi, tujuan dari wisata itu sendiri adalah untuk mencari hiburan dan keluar dari kepenatan aktivitas. Kota Semarang merupakan salah satu kota di Indonesia yang banyak dijadikan kota tujuan wisata, ditambah dengan kedudukannya sebagai Ibu Kota Provinsi Jawa Tengah, membuatnya banyak dikunjungi oleh wisatawan baik domestik maupun mancanegara. Tempat wisatanyapun beragam, dari yang hanya sebuah kawasan ramai ditengah kota, wisata alam, wisata kuliner, wisata religi serta wisata bersejarah. Lawang Sewu adalah objek wisata gedung kuno yang menjadi salah satu ikon wisata Kota Semarang. Pariwisata merupakan salah satu sektor yang berpengaruh terhadap pembangunan suatu daerah dan menjadi salah satu sumber pendapatan daerah. Karena sangat berpengaruhnya sektor pariwisata, maka untuk melihat perkembangan pariwisata yang ada di Kota Semarang diperlukan penelitian analisis, yang nantinya dapat digunakan untuk menentukan apa yang harus dilakukan untuk pengembangan dan mempertahankan objek wisata yang ada di Kota Semarang.

Dari latar belakang tersebut, penelitian ini akan menganalisis sentimen ulasan tentang pariwisata di Kota Semarang dengan objek penelitian adalah objek wisata Lawang Sewu, data ulasan di ambil dari kolom *review* pada situs TripAdvisor yang merupakan ulasan pengunjung objek wisata Lawang Sewu Kota Semarang, yang nantinya dapat memberikan informasi yang berguna untuk Dinas Kebudayaan dan Pariwisata Kota Semarang, pengelola objek wisata Lawang Sewu maupun pihak lain yang membutuhkan.

1.2. Rumusan Masalah

Berdasarkan latar belakang, maka permasalahan yang akan dikaji dalam penelitian ini adalah sebagai berikut.

1. Bagaimana gambaran umum data ulasan pengunjung objek wisata Lawang Sewu pada situs TripAdvisor?
2. Bagaimana penerapan metode klasifikasi *Support Vector Mechine*, *Naïve Bayes*, *Decision Tree, Random Forest* dan *Logistic Regression* dalam pengklasifikasian

sentimen positif dan sentimen negatif pada data ulasan pengunjung objek wisata Lawang Sewu?

3. Pengetahuan apa yang didapatkan dari analisis klasifikasi pada data ulasan pengunjung objek wisata Lawang Sewu?
4. Bagaimana visualisasi data dari hasil klasifikasi?

1.3. Batasan Masalah

Dalam penelitian ini akan fokus membahas pada analisis sentimen dengan batasan masalah seperti berikut.

1. Penelitian ini menggunakan data ulasan pengunjung objek wisata Lawang Sewu berbahasa Indonesia dan Inggris dari situs TripAdvisor yang tercatat mulai dari bulan Januai 2011 hingga Maret 2020.
2. Analisis yang digunakan dalam penelitian ini menggunakan metode klasifikasi *Support Vector Mechine*, *Naïve Bayes*, *Decision Tree*, *Random Forest*, dan *Logistic Regression*.
3. Dalam analisis ini digunakan bahasa pemrograman Python.
4. Pelabelan kelas sentimen berdasarkan kamus lexicon.

1.4. Tujuan Penelitian

Tujuan dari penelitian ini secara umum sebagai berikut.

1. Mengetahui gambaran umum data ulasan pengunjung objek wisata Lawang Sewu pada situs TripAdvisor.
2. Mengetahui penerapan metode klasifikasi *Support Vector Mechine, Naïve Bayes, Decision Tree, Random Forest* dan *Logistic Regression* dalam pengklasifikasian sentimen positif dan sentimen negatif pada data ulasan pengunjung objek wisata Lawang Sewu.
3. Mendapatkan pengetahuan dari analisis klasifikasi pada data ulasan pengunjung objek wisata Lawang Sewu.
4. Mengetahui visualisasi data dari hasil klasifikasi.

1.5. Manfaat Penelitian

Manfaat dari penelitian ini adalah sebagai berikut.

1. Bagi mahasiswa, menerapkan ilmu pengetauan yang telah diperoleh selama di bangku perkuliahan dan mengetahui masing-masing algoritma dalam melakukan klasifikasi data ulasan pengunjung objek wisata Lawang Sewu Kota Semarang.
2. Bagi Jurusan Matematika, dapat dijadikan bahan studi kasus dan acuan serta referensi bagi mahasiswa.
3. Bagi instansi, dapat dijadikan sebagai bahan evaluasi Dinas Kebudayaan dan Pariwisata Kota Semarang, pengelola objek wisata Lawang Sewu maupun pihak lain yang membutuhkan, agar kedepannya objek wisata Lawang Sewu dapat tetap lestari dan banyak dikunjungi oleh wisatawan baik domestik maupun mancanegara.

1.6. Sistematika Penulisan

Sistematika penulisan berguna untuk memudahkan dalam memahami jalan pemikiran secara keseluruhan tugas akhir. Secara garis besar tugas akhir ini dibagi menjadi tiga bagian, yakni :

1. Bagian awal

   Bagian ini terdiri atas halaman judul, pernyataan, pengesahan, motto dan persembahan, prakata, abstrak, daftar isi, daftar tabel, daftar bagan, daftar gambar dan daftar lampiran.
2. Bagian isi

   Bagian ini merupakan bagian laporan penelitian yang terdiri atas lima bab dengan rincian sebagai berikut.

   BAB I PENDAHULUAN

   Berisi latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penulisan.

   BAB II TINJAUAN PUSTAKA

   Berisi uraian teoritis dan materi yang mendasari pemecahan masalah-masalah yang berhubungan dengan pembahasan penelitian tugas akhir ini.

BAB III METODE PENELITIAN

Berisi metode penelitian yang digunakan dalam penelitian ini, diantaranya populasi dan sampel, variabel penelitian, metode pengumpulan data, analisis data dan alur penelitian.

BAB IV HASIL DAN PEMBAHASAN

Berisi hasil dan pembahasan dari permasalahan.

BAB V PENUTUP

Berisi simpulan dan saran dari hasil penelitian.

3. Bagian akhir

   Bagian ini terdiri dari daftar pustaka dan lampiran.

BAB II
TINJAUAN PUSTAKA

Pada tinjauan pustaka dijelaskan mengenai teori-teori dan literatur yang menjadi dasar dalam penyelesaian masalah pada penelitian ini. Berbagai sumber berupa buku, *e-book*, artikel dan jurnal digunakan untuk mendukung penyelesaian tugas akhir ini.

2.1. Pariwisata Kota Semarang

Menurut KBBI, pariwisata adalah kegiatan yang berhubungan dengan perjalanan untuk rekreasi, dan menurut Undang Undang No. 10 Tahun 2009 tentang kepariwisataan, pariwisata adalah berbagai macam kegiatan wisata dan didukung fasilitas serta layanan yang disediakan masyarakat setempat, sesama wisatawan, pemerintah daerah dan pengusaha. Menurut (Meyers 2009) berwisata adalah aktivitas perjalanan yang dilakukan oleh sementara waktu, dari tempat tinggal semula ke daerah tujuan, dengan alasan bukan untuk menetap atau mencari nafkah, melainkan hanya untuk memenuhi rasa ingin tahu, menghabiskan waktu senggang atau libur serta tujuan-tujuan lainnya.

Kota Semarang adalah Ibu Kota Provinsi Jawa Tengah, dengan kedudukannya sebagai Ibu Kota tentu Semarang memliki beragam fasilitas yang lengkap dan memadai, seperti bidang transportasi, kesehatan, pendidikan, perbelanjaan, dan lain-lain. Fasilitas-fasilitas ini mendukung perkembangan pariwisata di Kota Semarang, terutama bidang transportasi. Kota Semarang dapat dijangkau oleh daerah lain dengan mudah mengunakan berbagai sarana transportasi, baik transportasi darat, transportasi udara maupun transportasi laut. Kemudahan ini menjadi salah satu daya tarik wisatawan untuk berwisata ke Kota Semarang, baik wisatawan domestik maupun mancanegara.

Setya (2017) mengungkapkan bahwa Kota Semarang memanfaatkan sektor pariwisata sebagai sumber pendapatan dan potensi dalam bersaing dengan daerah lain. Alasannya adalah karena kekayaan potensi pariwisata di Kota Semarang yang

cukup banyak dan perlu di kembangkan. Pembangunan pariwisata Kota Semarang diperlukan untuk medorong pemerataan kesempatan usaha dan memperoleh manfaat serta mampu bersaing di lingkungan lokal sampai internasional.

Objek wisata adalah segala sesuatu yang memilik keunikan, keindahan dan nilai yang berupa keanekaragaman kekayaan alam, budaya, dan hasil buatan manusia yang menjadi sasaran atau tujuan kunjungan wisatawan. Objek wisata merupakan daya tarik agar orang-orang mau datang berkunjung ke daerah tujuan wisata. Objek wisata dapat berupa wisata sejarah, wisata religi, wisata kuliner, wisata alam dan lainnya. Menurut (Devy and Soemanto 2017) keberadaan objek dan daya tarik wisata merupakan mata rantai terpenting dalam suatu kegiatan pariwisata, hal ini disebabkan karena faktor utama yang membuat pengunjung atau wisatawan untuk mengunjungi daerah tujuan wisata adalah potensi dan daya tarik yang dimiliki objek wisata tersebut.

Kota Semarang memiliki wisata yang beragam, keberagaman ini menjadi daya tarik wisatawan, beberapa objek wisata yang ada di Kota Semarang diantaranya sebagai berikut.

1. Wisata alam: Kota Semarang memiliki banyak pilihan wisata alam, diantaranya Pantai Maron, Desa Wisata Lembah Kalipancur, Pantai Marina, pantai Tirang, Brown Canyon.
2. Wisata kuliner: Kota Semarang mempunyai makanan khas yaitu Lumpia, Wingko Babat dan Bandeng Presto.
3. Wisata religi: beberapa tempat wisata religi yang ada di Kota Semarang antara lain Masjid Agung Jawa Tengah, Gereja Blenduk, Klenteng Sam po Kong, Pura Agung Giri Natha dan Vihara Avalokitesvara.
4. Wisata sejarah: Kota Semarang memiliki banyak sejarah yang masih lestari hingga kini, beberapa diantaranya adalah Lawang Sewu, Tugu Mudan dan Kota Lama.

5. Wisata hiburan: banyak objek wisata buatan yang dibangun di Kota Semarang, beberapa wisata hiburan di Kota Semarang adalah Simpang Lima, Puri Maerokoco, Taman Margasatwa Semarag (Kebun Binatang Mangkang), Museum Ronggowarsito, dan Museum Mandala Bhakti.

2.2. Objek Wisata Lawang Sewu

Lawang sewu atau yang dalam bahasa Indonesia berarti seribu pintu, adalah gedung tua bersejarah yang terletak di Kota Semarang. Sebutan Lawang Sewu pada gedung ini berasal dari masyarakat setempat yang menyebutnya "Lawang Sewu", karena gedung ini memiliki pintu yang sangat banyak, meskipun sebenarnya jumlah pintu yang ada pada gedung Lawang Sewu tidak mencapai seribu. Gedung ini memiliki banyak jendela yang tinggi dan lebar, sehingga banyak masyarakat menganggapnya sebagai pintu.

Dinas Kebudayaan dan Pariwisata Kota Semarang (2019) pada situs resminya memaparkan, Lawang Sewu merupakan sebuah gedung bersejarah yang memiliki nilai historis sangat tinggi, berdiri tepat di pusat keramaian di jantung Kota Semarang, Jawa Tengah. Pada masa lalu, gedung ini merupakan kantor dari Nederlands Indische Spoorweg Maatschappij atau Jawatan Kereta Api. Dibangun pada tahun 1904 dan selesai pada tahun 1907. Terletak di kawasan bundaran Tugu Muda yang dahulu kala disebut Wilhelminaplein. Gedung kuno dan megah berlantai dua ini setelah masa kemerdekaan Indonesia dipakai sebagai kantor Djawatan Kereta Api Repoeblik Indonesia (DKARI) atau sekarang PT. Kereta Api Indonesia. Pernah dipakai sebagai Kantor Badan Prasarana Komando Daerah Militer (Kodam IV/Diponegoro) dan Kantor Wilayah (Kanwil) Kementerian Perhubungan Jawa Tengah.

Pada masa perjuangan gedung Lawang Sewu memiliki catatan sejarah tersendiri yaitu ketika berlangsung peristiwa Pertempuran Lima Hari di Semarang (14 Oktober - 19 Oktober 1945). Gedung tua ini menjadi lokasi pertempuran yang hebat antara pemuda AMKA atau Angkatan Muda Kereta Api melawan Kempetai dan Kidobutai, Jepang. Pemerintah Kota Semarang memasukkan Lawang Sewu sebagai

salah satu dari 102 gedung kuno atau bersejarah di Kota Semarang yang sangat dilindungi. Kini, gedung tua ini telah mengalami tahap konservasi dan revitalisasi yang dilakukan oleh Unit pelestarian benda dan bangunan bersejarah PT Kereta Api Persero dan dijadikan sebagai objek wisata yang ramai dikunjungi turis lokal, nasional, serta internasional.

2.3. Ulasan Pengunjung pada Situs TripAdvisor

TripAdvisor merupakan komunitas wisata terbesar di dunia untuk mencari informasi, saran dan opini dari jutaan wisatawan, yang berguna untuk membantu para pengguna lain dalam merencanakan perjalanan wisata. *Website* ini berupa komunitas yang ramah dan santai yang berisi percakapan yang jujur di antara para wisatawan, dapat dilihat bahwa TripAdvisor adalah sebuah alamat *web* yang dirancang khusus bagi para wisatawan, yang bertujuan untuk memudahkan penggunanya dalam menemukan informasi mengenai tempat wisata yang ingin dikunjunginya yang berupa penginapan, makanan, dan tempat hiburan.

Afifah (2015) menjelaskan bahwa situs ini memungkinkan setiap penggunanya untuk mengeksperikan pendapat, pengalamannya, ataupun tips mengenai perjalanan mereka. Hal ini dapat dilihat oleh pengguna lain yang ingin mendapatkan rujukan tujuan perjalanan mereka berupa hotel atau restoran yang ingin mereka sewa. Situs ini memungkinkan setiap orang untuk mengekspresikan diri, berbagi pengalaman dan memberikan tips mengenai perjalanan mereka. Dengan menggunakan situs ini setiap orang bebas untuk memberikan tanggapan dan memberikan ulasan berdasarkan pengalaman meraka, baik ulasan itu berupa ulasan negatif maupun ulasan yang bersifat positif. Banyaknya ulasan pada situs ini secara tidak langsung akan mempengaruhi pelanggan ataupun calon pelanggan dalam memilih dan memutuskan rencana perjalanan wisata. Ulasan-ulasan pada situs tersebut juga secara tidak langsung akan memberikan pengaruh pada citra dan merek objek tersebut.

Dalam TripAdvisor terdapat berbagi fitur yang dapat mempermudah para wisatawan diantaranya, rekomendasi hotel, penginapan dan restoran, booking tiket

penerbangan, dan lainnya. Tripadvisor membantu pemesan dan membandingkan harga rendah dari lebih 200 situs pemesanan diseluruh dunia untuk memesan hotel, penerbangan, penyewaan rumah & apartemen liburan, dan restoran. Tripadvisor telah diakses lebih dari 859 juta dari wisatawan (TripAdvisor 2018).

TripAdvisor merupakan bagian dari TripAdvisor Media Group yang dimiliki dan dioperasikan oleh Trip Advisor LLC. Situs ini beroperasi di 48 negara di seluruh dunia. TripAdvisor memiliki lambang burung hantu dengan mata merah hija, burung hantu melambangakan TripAdvisor mampu melihat ke segala tempat, dengan kemampuan penglihatannya yang tajam serta gerak kepala yang luas, dan warna merah serta hijau pada matanya sebagai simbol kemana wisatawan pergi (hijau) dan tidak pergi (merah). Awalya TripAdvisor merupakan situs B2B (*business to business*), tujuanya mempertemukan produsen dengan produsen, namun kemudian dimanfaatkan komunitas *traveller* untuk berbagi pengalaman, yang akhirnya berubah menjadi situs *review* perjalanan wisata terbesar di dunia. Postingan *review* wisata pertama dilakukan pada tahun 2001.

Gambar 2.1 Logo TripAdvisor
(Sumber: www.tripadvisor.com)

2.4. Pengumpulan Data dengan Web Scraping

Saat ini pengumpulan data dari internet menjadi sebuah tren, dan bahkan sudah sangat umum dilakukan oleh perusahaan atau individu untuk kepentingan tertentu. *Web scraping* yaitu proses ekstrasi data dari sebuah *website*, mengambil data berbentuk teks yang umumnya bertipe HTML atau XHTML. *Web scraping* dapat

membantu untuk mempermudah proses pengumpulan data, apalagi untuk data yang berjumlah besar dari suatu website.

Menurut (Josi, Abdillah, and Suryayusra 2014) *web scraping* atau sering dikenal sebagai *screen scraping*, tidak dapat dimasukkan dalam bidang *data mining* karena *data mining* menyiratkan upaya untuk memahami pola semantik atau tren dari sejumlah besar data yang telah diperoleh. Aplikasi *web scraping* hanya fokus pada cara memperoleh data melalui pengambilan dan ekstraksi data dengan ukuran data yang bervariasi. Ada beberapa teknik automasi yang bisa dilakukan untuk melakukan *web scraping* diantaranya sebagai berikut.

1. Manual: proses *web scraping* yang paling sederhana adalah menyalin data *website* secara manual. Karena harus mengambil dan menyimpan informasi yang diperlukan satu per satu, teknik ini memakan waktu lama. Akan tetapi, metode ini paling efektif dari segi pencarian data. Tidak seperti *too*l atau *bot*, sudah diketahui letak informasi yang ingin disalin dari suatu *website*, dengan demikian hasil *web scraping* dengan cara ini sangat akurat.
2. *Regular expression*: *regular expression* adalah baris kode yang digunakan dalam algoritma pencarian untuk menemukan tipe data tertentu dari sebuah file. Dalam konteks *web scraping*, file yang dimaksud adalah file-file penunjang sebuah *website*. Keuntungan utama menggunakan *regular expression* untuk *web scraping* adalah konsistensi *syntax*nya di dalam berbagai bahasa pemrograman. Oleh karena itu, teknik ini sangat fleksibel. Ditambah lagi, *regular expression* dapat digunakan untuk mencari data berdasarkan jenisnya, seperti nama produk, harga, dan alamat email.
3. *Parsing* HTML: salah satu teknik yang paling banyak digunakan dalam *web scraping*. Biasanya parsing HTML dilakukan melalui JavaScript dan menarget halaman HTML linear dan nested. Metode yang cepat ini mengidentifikasi *script* HTML dari website, yang mungkin saja dilakuakan secara manual sebelumnya. *Script* ini kemudian digunakan untuk mengekstraksi teks, *link* dan data.
4. *Parsing* DOM: DOM adalah singkatan dari *Document Object Model*. *Scrapers* yang ingin mengetahui cara kerja internal halaman web dan mengekstrak *script*

yang berjalan di dalamnya biasa memilih untuk melakukan *web scraping* melalui parsing DOM.

5. Xpath: XML Path atau lebih sering disebut Xpath adalah bahasa *query* yang bekerja di dokumen XML. Karena dokumen XML biasa disusun dengan struktur pohon (*tree structure*), Xpath bisa digunakan untuk menavigasi struktur dokumen tersebut dengan memilih *nodes* berdasarkan berbagai parameter.
6. Google Sheet: Google Sheets juga dapat digunakan sebagai alat *scraping*, pada Google Sheets dapat memanfaatkan fungsi IMPORTXML untuk melakukan *scraping* data dari *website*. Selain itu, juga dapat menggunakan *command* ini untuk melihat apakah *website* yang kita miliki aman dari *scraping*.

2.5. Analisis Deskriptif

Analisis deskriptif menurut (Sugiyono 2009) adalah metode yang berfungsi untuk mendeskripsikan atau memberi gambaran terhadap objek yang diteliti melalui data atau sampel yang telah terkumpul sebagimana adanya, tanpa melakukun analisis dan membuat kesimpulan yang berlaku umum.

Analisis data dengan menerapkan metode deskriptif dinyatakan sebagai analisis statistik sederhana atau yang paling sederhana. Akan tetapi, hasil analisis statistik deskriptif tersebut dapat menjadi masukan yang sangat berharga untuk mengambil keputusan, tergantung pada bentuk dan cara menyajikan hasil analisis tersebut (Agung 2000).

2.6. Text Mining

Data mining atau penambangan data merupakan gabungan sejumlah disiplin ilmu komputer yang didefinisikan sebagai proses penemuan pola-pola baru dari kumpulan-kumpulan data yang sangat besar, meliputi metode-metode yang merupakan irisan dari *artificial intelligence, machine learning, statistics dan database systems.* Menurut (Suyanto 2019) *data mining* ditunjuk untuk mengekstrak (mengambil intisari) pengetahuan dari sekumpulan data sehingga didapatkan struktur yang dapat dimengerti manusia, serta meliputi basis data dan managemen data,

prapemrosesan data, pertimbangan model, dan inferensi, ukuran keterikatan, pertimbangan kompleksitas pasca pemrosesan terhadap struktur yang ditemukan, visualisasi, dan *online updating*.

Kegunaan *data mining* dapat dibagi menjadi dua, yaitu deskriptif dan prediktif. Deskriptif berarti *data mining* digunakan untuk mencari pola-pola yang dapat dipahami manusia yang menjelaskan karakteristik data. Sedangkan prediktif berarti *data mining* digunakan untuk membentuk sebuah model pengetahuan yang akan digunakan untuk melakukan prediksi. Berdasar fungsionalitasnya, tugas-tugas data mining dapat dikelompokan ke dalam enam kelompok berikut.

1. Klasifikasi (*classification*): mengeneralisasi struktur yang diketahui untuk diaplikasikan pada data-data baru.
2. Klasterisasi (*clustering*): mengelompokan data yang tidak diketahui label kelasnya ke dalam sejumlah kelompok tertentu sesuai dengan ukuran kemiripannya.
3. Regresi (*regression*): menentukan suatu fungsi yang memodelkan data dengan galat (kesalahan prediksi) seminimal mungkin.
4. Deteksi anomali (*anomaly detectioni*): mengidentifikasi data yang tidak umum, bisa berupa *outlier* (pencilan), perubahan ataudeviasi yang mungkin sangat penting dan perlu investigasi lebih lanjut.
5. Pembelajaan aturan asosiasi (*assosiation rule learning*) atau pemodelan kebergantungan (*dependency modeling*): mencari relasi antar variabel.
6. Perangkuman (*summarization*): menyediakan representasi data yang lebih sederhana, meliputi visualisasi dan pembuatan laporan.

Text mining adalah penambangan data yang berupa teks, tujuan dari *text mining* adalah untuk mendapatkan informasi yang berguna dari sekumpulan dokumen. Menurut (Purbo 2019) *text mining* juga dapat diartikan sebagai proses untuk memperoleh, menemukan (discovering), memvisualisasikan (presenting), mengevaluasi pengetauan dari kumpulan besar teks dokumen.

Text mining merupakan penerapan konsep dan teknik *data mining* untuk mencari pola dalam teks, yaitu proses penganalisisan teks guna mencari informasi

yang bermanfaat untuk tujuan tertentu. Secara umum ada empat proses yang perlu dijalankan dalam *text mining* yaitu sebagai berikut.

1. Akusisi data: pengumpulan data.
2. *Text preprocessing*: penyiapan data atau pembersihan data, karena pada umumnya data yang didapat pada *text mining* adalah *unstructured* data.
3. Pemodelan yang akan melalui proses *looping* dengan proses evaluasi dan validasi.
4. Presentasi dan interaksi biasanya dilakukan untuk memvisualisasikan hasil pemodelan yang dilakukan.

2.7. Pembobotan Kata TF-IDF

Pembobotan kata (*term weighting*) adalah proses pemberian bobot *term* pada dokumen. Pembobotan dasar dilakukan dengan menghitung frekuensi kemunculan *term* dalam dokumen. Frekuensi kemunculan (*term frequency*) merupakan gambaran sejauh mana *term* tersebut mewakili isi dokumen. Semakin besar kemunculan suatu *term* dalam dokumen akan memberikan nilai kesesuian yang semakin besar. Pembobotan dapat mencerminkan betapa pentingnya sebuah kata dalam dokumen. Yunus (2020) menjelaskan, pembobotan ini nantinya digunakan oleh algoritma *machine learning* untuk klasifikasi dokumen.

1. *Term Frequency–Inverse Document Frequency* (TF-IDF) adalah bobot *term* terhadap dokumen. TF-IDF dirumuskan sebagai berikut.
   $w_{ij} = tf_{ij} \times idf_j$
2. *Term Frequency* (TF) adalah jumlah kemunculan *term* dalam dokumen. Pada *Term Frequency* (TF), terdapat beberapa jenis formula yang dapat digunakan.
   a. TF biner (*binary* TF): hanya memperhatikan jumlah kemunculan suatu *term* ada atau tidak dalam dokumen, jika ada diberi nilai satu (1), jika tidak ada diberi nilai nol (0).

b. TF murni (*raw* TF): nilai TF dihitung berdasarkan jumlah kemunculan suatu t*erm* di dokumen, jika *term* tersebut muncul sebanyak tiga kali dalam dokumen maka TF bernilai tiga.

c. TF logaritmik: perhitungan TF logaritmik digunakan untuk menghindari dominasi dokumen yang mengandung sedikit *term* dalam *query*, namun mempunyai frekuensi yang tinggi.

$$tf_{ij} = 1 + \log(tf)$$

d. TF normalisasi: perhitungan TF normalisasi menggunakan perbandingan antara frekuensi term dengan nilai maksimum dari keseluruhan atau kumpulan frekuensi term yang ada pada suatu dokumen.

$$tf_{ij} = 0{,}5 + 0{,}5\ x\ \frac{tf}{\max tf}$$

3. *Inverse Document Frequency* (IDF) berfungsi mengurangi bobot suatu *term* jika kemunculannya banyak tersebar diseluruh dokumen. Perhitungan IDF diperlukan, karena jika hanya menggunakan TF saja dikhawatirkan akan muncul kata umum yang akan dominan, yang seharusnya kata tersebut dihilangkan. IDF dirumuskan sebagai berikut.

$$idf_j = log\left(\frac{D}{df_j}\right)$$

Di mana,

D : jumlah semua dokumen yang ada dalam data

df_j : jumlah dokumen yang mengandung *term*

2.8. Analisis Sentimen

Menurut (Rozi, Pramono, and Dahlan 2012) analisis sentimen atau *opinion mining* merupakan proses memahami, mengekstrak dan mengolah data tekstual secara otomatis untuk mendapatkan informasi sentimen yang terkandung dalam suatu kalimat opini. Analisis sentimen dilakukan untuk melihat pendapat atau kecenderungan opini terhadap sebuah masalah atau objek oleh seseorang, apakah

cenderung berpandangan (beropini) negatif atau positif. Berdasarkan sumber datanya, analisis sentimen dapat dibedakan menjadi dua jenis, yaitu.

1. *Coarse grained Sentiment Analysis* : analisis sentimen yang dilakukan pada level dokumen. Secara garis besar fokus utama dari analisis sentimen jenis ini adalah menganggap seluruh isi dokumen sebagai sebuah sentimen positif atau sentimen negatif.
2. *Fined grained Sentiment Analysis* : analisis sentimen yang dilakuakan pada level kalimat. Fokus utama *fined-grained* adalah menentukan sentimen pada setiap kalimat suatu dokumen, dimana kemungkinan yang terjadi adalah terdapat sentimen pada level kalimat yang berbeda pada suatu dokumen.

VADER (*Valence Aware Dictionary and sEntiment Reasoner*) adalah metode analisis sentimen yang dikembangkan oleh Hutto dan Gilbert tahun 2014. Metode ini mengambil kalimat sebagai input dan memberikan nilai persen untuk positif, negatif dan netral serta compound (polaritas keseluruhan kalimat). Compound adalah metrik yang menghitung jumlah semua peringkat lexicon yang telah dinormalisasi antara -1 (paling negatif) dan +1 (paling positif) (Alif 2020).

2.9. **Metode Klasifikasi**

Menurut (Suyanto 2019) klasifikasi adalah bagaimana mempelajari sekumpulan data sehingga dihasilkan aturan yang bisa mengklasifikasi atau mengenali data-data baru yang belum pernah dipelajari. Klasifikasi adalah proses untuk menyatakan suatu objek data sebagai salah satu kategori (kelas) yang telah didefinisikan sebelumnya.

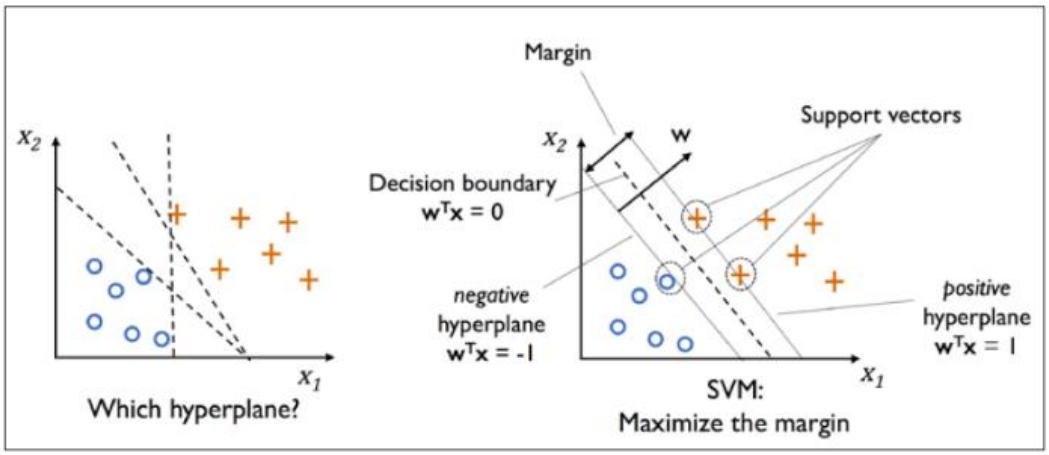
Model klasifikasi dapat dibangun dengan berdasarkan pengetahuan seorang pakar. Namun, mengingat himpunan data yang relatif besar, model klasifikasi lebih sering dibangun menggunakan teknik pembelajaran dalam bidang *mechine learning.* Proses pembelajaran secara otomatis terhadap suatu himpunan data mampu menghasilkan model klasifikasi (fungsi target) yang memetakan objek data *x* (input) ke salah satu kelas *y* yang telah didefinisikan sebelumnya. Jadi, proses pembelajaran memerlukan masukan (input) berupa himpunann data latih (*training*) yang yang

berlabel (memiliki atribut kelas) dan mengeluarkan *output* yang berupa sebuah model klasifikasi. Analisis sentimen pada level kalimat (*Fined grained Sentiment Analysis*) dapat dianalisis menggunakan metode klasifikasi apa saja, untuk menentukan metode mana yang paling baik, dilakukan dengan membandingkan nilai akurasinya. Semakin besar tingkat pengenalan (akurasi) maka semakin baik metode klasifikasi tersebut dalam mengklasifikasikan data. Metode klasifikasi memiliki beberapa macam algoritma diantaranya *Support Vector Mechine, Naïve Bayes, Decision Tree, Random Forest* dan *Logistic Regression.*

2.9.1. Support Vector Mechine

Support Vector Machine (SVM) diperkenalkan oleh Vapnik pada tahun 1992 sebagai suatu teknik klasifikasi yang efisien. SVM memiliki konsep yang jauh lebih matang, lebih jelas secara matematis dibanding teknik-teknik klasifikasi sebelum era 1990an (Suyanto 2019).

SVM digunakan untuk mencari *hyperplane* terbaik dengan memaksimalkan jarak antar kelas. H*yperplane* adalah sebuah fungsi yang dapat digunakan untuk pemisah antar kelas. Dalam dua dimensi fungsi yang digunakan untuk klasifikasi antar kelas disebut sebagai *line whereas,* fungsi yang digunakan untuk klasifikasi antas kelas dalam tiga dimensi disebut *plane similarly,* sedangan fungsi yang digunakan untuk klasifikasi di dalam ruang kelas dimensi yang lebih tinggi di sebut *hyperplane*. Dalam SVM objek data terluar yang paling dekat dengan *hyperplane* disebut *support vector*. Objek yang disebut *support vector* paling sulit diklasifikasikan dikarenakan posisi yang hampir tumpang tindih (*overlap*) dengan kelas lain. Mengingat sifatnya yang kritis, hanya *support vector* inilah yang diperhitungkan untuk menemukan *hyperplane* yang paling optimal oleh SVM.



Gambar 2.2 *Hyperplane* pada Algoritma *Support Vector Mechine*
(Sumber: www.medium.com)

2.9.2. Naïve Bayes

Naïve bayes classifier merupakan teknik klasifikasi berdasarkan teorema *bayes* dengan asumsi independensi di antara para prediktor. Klasifikasi *naïve bayes* memprediksi peluang di masa depan berdasarkan pengalaman di masa sebelumnya (Imron 2019).

Handayani dan Pribadi (2015) menjelaskan bahwa metode klasifikasi *naïve bayes* dapat memprediksi probabilitas keanggotaan kelas suatu data tuple yang akan masuk ke dalam kelas tertentu, sesuai dengan perhitungan probabilitas. Keuntungan penggunan adalah bahwa metode ini hanya membutuhkan jumlah data pelatihan (*training* data) yang kecil untuk menentukan estimasi parameter yang diperlukan dalam proses pengklasifikasian. Karena yang diasumsikan sebagai variabel independent, maka hanya varians dari suatu variabel dalam sebuah kelas yang dibutuhkan untuk menentukan klasifikasi, bukan keseluruhan dari matriks kovarians.

2.9.3. Decision Tree

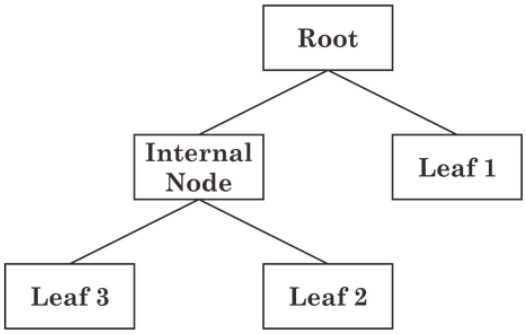
Menurut (Vulandari 2017) d*ecision tree* adalah salah satu metode klasifikasi yang paling populer, karena mudah untuk diinterpretasi oleh manusia. *Decision tree* adalah model prediksi menggunakan struktur pohon atau struktur berhirarki. Konsep dari *decision tree* adalah mengubah data menjadi pohon keputusan dan aturan-aturan keputusan. Manfaat utama dari penggunaan *decision tree* adalah

kemampuannya untuk mem*break down* proses pengambilan keputusan yang kompleks menjadi lebih simpel, sehingga pengambilan keputusan akan lebih menginterpretasikan solusi dari permasalahan.

Pohon keputusan adalah sebuah struktur yang dapat digunakan untuk membagi kumpulan-kumpulan data yang besar menjadi himpunan-himpunan *record* yang lebih kecil dengan menerapkan serangkaian aturan keputusan. Dengan masing-masing rangkaian pembagi, anggota himpunan hasil menjadi mirip satu dengan yang lainnya. Data dalam pohon keputusan biasanya dinyatakan dalam bentuk tabel dengan atribut dan *record*. Atribut menyatakan suatu parameter yang dibuat sebagai kriteria dalam pembentukan pohon (Mardi 2017).

Decision tree dinamakan juga pohon keputusan, karena aturan yang terbentuk mirip dengan bentuk pohon. Konsep dasar dari pohon keputusan adalah mengubah data menjadi model pohon keputusan, kemudian mengubah model pohon menjadi *rule* dan menyederhanakan *rule*. Data dalam pohon keputusan dinyatakan dalam bentuk tabel dengan atribut dan *record*. Menurut (Setio, Saputro, and Winarno 2020) pohon keputusan memiliki *node* pohon yang merepresentasikan atribut yang telah diuji dan setiap cabangnya merupakan suatu pembagian hasil uji serta *node* daun (leaf) merepresentasikan kelompok kelas tertentu. Level *node* teratas dari sebuah pohon keputusan adalah *node* akar (*root*) yang biasanya berupa atribut yang memiliki pengaruh paling besar pada suatu kelas tertentu. Kontruksi pada pohon keputusan terdapat tiga jenis node, yaitu.

1. Akar (*root*): merupakan *node* teratas, pada node ini tidak ada input dan dapat mempunyai *output* lebih dari satu.
2. Internal node: merupakan node percabangan, pada node ini hanya terdapat satu input dan mempunyai *output* minimal dua.
3. Daun (*leaf*): merupakan node akhir atau terminal node, pada node ini hanya terdapat satu input dan tidak mempunyai *output* (simpul terminal).



Gambar 2.3 *Decision Tree*
(Sumber: www.medium.com)

2.9.4. Random Forest

Suyanto (2019) menjelaskan bahwa *random forest* atau yang sering juga disebut dengan *random ensembles* adalah kombinasi pohon keputusan sedemikian hingga setiap pohon bergantung pada nilai-nilai vektor acak yang disampling secara independen dengan distribusi yang sama. *Random forest* menggunakan seleksi fitur acak untuk memilah setiap simpul sehingga mampu menghasilkan akurasi relatif tinggi. *Random forest* dibangun berdasarkan pohon keputusan dengan menggabungkan pohon-pohon sederhana untuk menghasilkan tingkat akurasi yang lebih baik.

2.9.5. Logistic Regression

Regresi logistik merupakan salah satu metode klasifikasi yang sering digunakan, regresi logistik merupakan suatu teknik analisis data dalam statistika yang bertujuan untuk mengetahui hubungan antara beberapa variabel, dimana variabel responnya adalah bersifat kategorik, baik nominal maupun ordinal dengan variabel penjelasnya dapat bersifat kategorik atau kontinu. Regresi logistik biner digunakan saat variabel respon merupakan variabel dikotomus (kategorik dengan 2 macam kategori), sedangkan regresi logistik multinomial digunakan saat variabel respon adalah variabel kategorik dengan lebih dari 2 kategori (Ramli, Yuniarti, and Goejantoro 2013).

2.10. Confusion Matrix

Rahman (2017) menjelaskan, *Confusion matrix* adalah metode yang digunakan untuk melakukan perhitungan akurasi ada konsep *data mining*. *Confusion matrix* digambarkan dengan tabel yang menyatakan jumlah data uji yang benar diklasifikasikan dan jumlah data uji yang salah diklasifikasikan. Menarianti (2015) memaparkan, urutan pengujian ditabulasikan dalam *confusion matrix* dimana kelas yang diprediksi ditampilkan di bagian atas matriks dan kelas yang diamati di bagian kiri. Setiap sel berisi angka yang menunjukkan berapa banyak kasus yang sebenarnya dari kelas yang diamati untuk diprediksi.

1. *True Positive* (TP): Merupakan jumlah data positif yang diprediksi benar oleh *classifier.*
2. *True Negative* (TN): Merupakan jumlah data negatif yang diprediksi benar oleh *classifier.*
3. *False Postive* (FP): Merupakan jumlah data negatif yang diprediksi salah oleh *classifier* (diprediksi sebagai data positif).
4. *False Negative* (FN): Merupakan jumlah data positif yang diprediksi salah oleh *classifier* (diprediksi sebagai data negatif).

Tabel 2.1 *Confusion Matrix*

Aktual	**Prediktif**	
	Negatif	**Positif**
Negatif	TN (*True Negative*)	FP (*False Positive*)
Positif	FN (*False Negative*)	TP (*True Positive*)

Evaluasi terhadap suatu *classifier* umumnya dilakukan menggunakan sebuah himpunan data uji, yang tidak digunakan dalam pelatihan *classifier* tersebut, dengan suatu ukuran tertentu. Terdapat sejumlah ukuran yang dapat digunakan untuk menilai atau mengevaluasi model klasifikasi, diantaranya sebagai berikut.

1. Akurasi (tingkat pengenalan): menyatakan persentase jumlah tuple dalam data uji yang diklasifikasikan dengan benar oleh *classifer*, dengan rumus

$$akurasi = \frac{TP + TN}{TP + TN + FP + FN}$$

2. *Error rate/misclassification rate* (tingkat kesalahan): menyatakan presentase jumlah tuple dalam data uji yang salah dalam pengklasifikasian, dengan rumus

$$error\ rate = \frac{FP + FN}{FP + FN + TP + TN}$$

3. *Recall/sensitivity*: dikenal juga sebagai *true positive rate*, yaitu tingkat pengenalan terhadap tuple positif, seberapa besar porsi tuple positif yang diklasifikasikan dengan benar, yang dirumuskan dengan

$$recall = \frac{TP}{TP + FN}$$

4. *Specificity*: dikenal juga sebagai *true negative rate*, yaitu tingkat pengenalan terhadap tuple negatif, seberapa besar porsi tuple negatif yang diklasifikasikan dengan benar, yang dirumuskan dengan

$$specificity = \frac{TN}{FP + TN}$$

5. *Precision* (ukuran kepastian): berapa presentase tuple yang dilabeli sebagai positif adalah benar pada kenyataannya, yang dirumuskan sebagai

$$precision = \frac{TP}{TP + FP}$$

2.11. Bahasa Pemrograman Python

Python merupakan bahasa pemrograman tingkat tinggi. Hal ini disebabkan karena kode yang dituliskan akan di *compil*e menjadi *byte code* dan dieksekusi sehingga Python cocok digunakan untuk *scripting language*, aplikasi *web* dan lain sebagainya (Suryono, Utami, and Luthfi 2018).

Sugiana (2003) menjelaskan, Python pertama kali dikembangkan oleh Guido van Rossum pada tahun 1990 di CWI, Belanda. Bahasa ini dikategorikan sebagai bahasa pemrograman tingkat tinggi (*very high level language*) dan juga merupakan bahasa berorientasi objek yang dinamis (*object oriented dynamic language*). Python diklaim sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan

sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif. Beberapa fitur yang dimiliki Phyton adalah.

1. Memiliki kepustakaan yang luas, Phyton telah menyediakan modul-modul siap pakai untuk berbagai keperluan.
2. Memiliki tata bahasa yang jernih dan mudah diperlajari.
3. Memiliki aturan layout kode sumber yang memudahkan pengecekan, pembacaan kembali dan penulisan ulang kode sumber berorientasi objek.
4. Memiliki sistem pengelolaan memori otomatis (*garbage collection*, seperti java) Modular, mudah dikembangkan dengan menciptakan modul-modul baru.
5. Memiliki fasilitas pengumpulan sampah otomatis, seperti halnya pada bahasa pemrograman Java, Phyton memiliki fasilitas pengaturan penggunaan ingatan komputer sehingga para pemrogram tidak perlu melakukan pengaturan ingatan komputer secara langsung.
6. Memiliki banyak fasilitas pendukung sehingga mudah dalam pengoprasiannya.

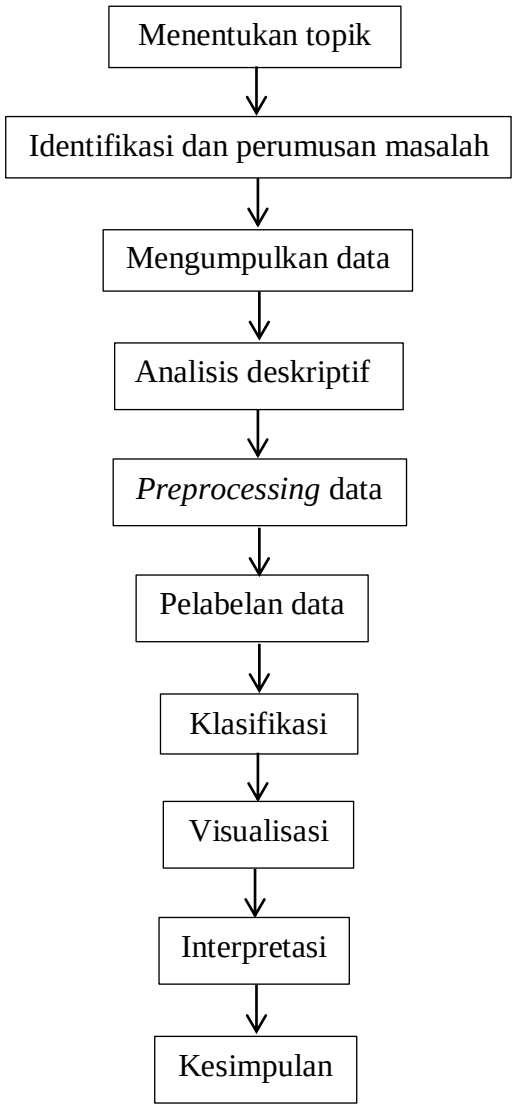
Saat ini kode Python dapat dijalankan di berbagai *platform* sistem operasi, beberapa di anataranya adalah Linux, Windows, Mac OS dan Java Virtual Machine.

2.12. Kerangka Berpikir

Dalam penelitian ini akan dilakukan tahap-tahap untuk menganalisis data yang diperoleh sebagai berikut.

1. Menentukan topik tugas akhir.
2. Mengidentifikasi permasalahan.
3. Mengumpulkan data ulasan pengunjung objek wisata Lawang Sewu pada situs TripAdvisor untuk dianalisis.
4. Analisis deskriptif untuk mendapatkan gambaran umum dari data ulasan.
5. Preprocessing data (pembersihan) sebelum data dianalisis.
6. Pelabelan data menjadi kelas sentimen positif dan negatif.
7. Analisis sentimen dengan menggunakan metode klasifikasi *Support Vector Mechine, Naïve Bayes, Decision Tree, Random Forest dan Logistic Regression.*
8. Memvisualisasikan hasil klasifikasi dengan *word cloud.*

9. Menginterpretasi hasil analisis.
10. Menarik kesimpulan.



Bagan 2.1 Kerangka Berpikir

BAB III
METODE PENELITIAN

3.1. Metode Pengumpulan Data

Pengumpulan data dilakukan dengan cara mengambil data ulasan pengunjung objek wisata Lawang Sewu dari kolom komentar situs TripAdvisor dengan cara *web scraping* menggunakan *package* Selenium dengan bahasa pemrograman Python.

3.2. Populasi dan Sampel Data

Populasi adalah wilayah generelisasi yang terdiri atas objek yang mempunyai kualitas dan karakteristik tertentu yang ditetapkan oleh peneliti untuk dipelajari dan kemudian ditarik kesimpulannya (Sugiyono 2009).

Populasi dalam penelitian ini adalah semua data ulasan objek wisata Lawang Sewu Kota Semarang yang terdapat pada situs TripAdvisor, yang tercatat sejak Januari 2011 hingga Maret 2020. Data populasi ini terdiri dari berbagai macam bahasa, seperti bahasa Indonesia, Inggris dan bahasa asing lainnya, diantaranya bahasa China, Belanda, Korea, Jepang, Prancis dan Spanyol.

Sampel adalah bagian dari suatu populasi, sampel yang digunakan dalam penelitian ini adalah data ulasan pengunjung Lawang Sewu Kota Semarang berbahasa Indonesia yang berjumlah 1303 ulasan dan data ulasan pengunjung Lawang Sewu Kota Semarang berbahasa Inggris yang berjumlah 652 ulasan.

3.3. Variabel Penelitian

Variabel penelitian pada dasarnya adalah segala sesuatu yang berbentuk apa saja yang ditetapkan oleh peneliti untuk dipelajari sehingga diperoleh informasi tentang hal tersebut, kemudian ditarik kesimpulannya. Dinamakan variabel karena ada variasinya (Sugiyono 2009). Dalam penelitian ini terdapat beberapa variabel diantaranya seperti berikut.

1. Rating/Penilaian: penilaian pengunjung terhadap objek wisata lawang Sewu, penilaian ini bernilai 1 sampai dengan 5.

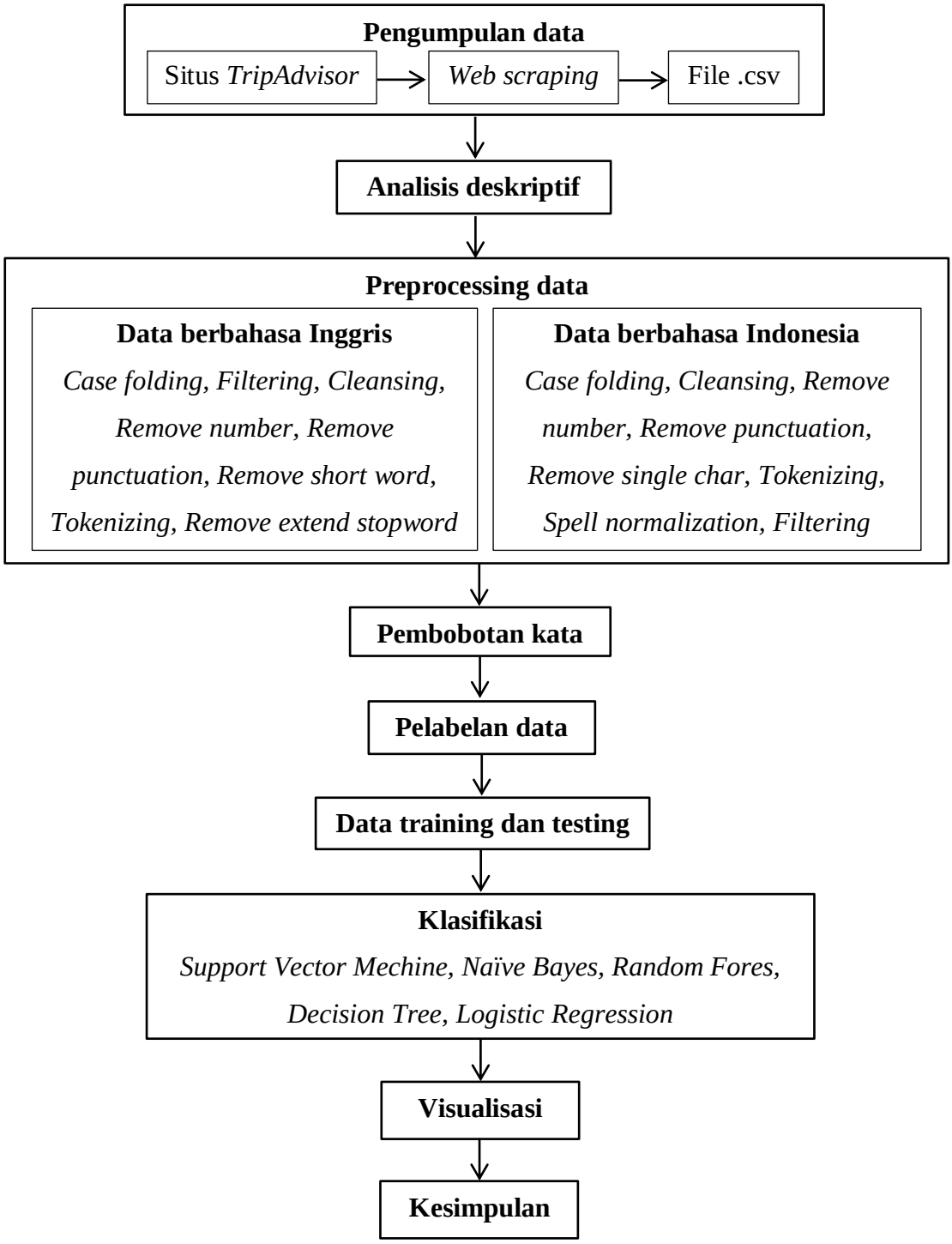
2. Name/Nama: *username* para pengunjung objek wisata Lawang Sewu yang menuliskan ulasan atau *review* pada situs TripAdvisor.
3. Review/Ulasan: deskripsi isi ulasan pengunjung objek wisata Lawang Sewu.

3.4. Analisis Data

Metode analisis yang digunakan dalam penelitian ini antara lain sebagai berikut.

1. Analisis Deskriptif: analisis ini bertujuan untuk mengetahui gambaran secara umum tentang data ulasan pengunjung objek wisata Lawang Sewu pada situs TripAdvisor.
2. *Preprocessin*g Data: tahap ini bertujuan untuk pembersihan data agar data dapat diolah.
3. Klasifikasi: analisis ini bertujuan untuk mengklasifikasikan data ulasan kedalam kelas sentimen positif atau negatif.
4. Visualisasi *Word Cloud*: analisis ini bertujuan untuk peritungan frekuensi kata untuk melihat topik apa yang paling banyak muncul dalam data ulasan pengunjung objek wisata Lawang Sewu pada situs TripAdvisor.

3.5. Alur Penelitian



Bagan 3.1 Alur Penelitian

BAB IV
HASIL DAN PEMBAHASAN

4.1 Hasil Penelitian

Hasil analisis dalam penelitian ini dilakukan dengan menggunakan bantuan *software* Anaconda Navigator dengan *tools* Jupyter Notebook dan bahasa pemrograman Python, dengan tujuan mengetahui penerapan analisis sentimen dengan menggunakan metode klasifikasi *Support Vector Mechine, Naïve Bayes, Random Forest, Decision Tree* dan *Logistic Regression* untuk mengetahui sentimen para pengunjung objek wisata Lawang Sewu Kota Semarang.

4.1.1 Data

Data dalam penelitian ini diperoleh dari kolom komentar situs TripAdvisor dengan cara *web scraping* menggunakan bahasa pemrograman Python. Data ulasan Lawang Sewu berbahasa Inggris dapat diperoleh dari situs web www.tripadvisor.com dengan nama "Lawang Sewu Building", data yang akan diambil adalah data rating, name dan review. Data ulasan berbahasa Indonesia dapat diperoleh dari situs web www.tripadvisor.co.id dengan nama "Gedung Lawang Sewu", data yang akan diambil adalah data penilaian, nama dan ulasan. Jumlah data yang diperoleh dari proses *web scraping*, untuk data ulasan berbahasa Inggris sebanyak 652 ulasan dan untuk data ulasan berbahasa Indonesia sebanyak 1303 ulasan. Contoh data ulasan yang didapat dari hasil *web scraping* dapat dilihat pada Tabel 4.1 untuk data ulasan berbahsa Inggris dan Tabel 4.2 untuk data ulasan berbahasa Indonesia.

Tabel 4.1 Data Ulasan Berbahasa Inggris

No.	Rating	Name	Review
1	4.0	9999stephenh	This is an amazing building (collection of buildings, actually) that has been very well restored. I stopped taking photos & videos only when storage space had run out and the battery was nearly flat! I visited at night - time but - despite Lawang Sewu's reputation - I didn't spot any ghosts!
2	4.0	deddy p	Want to come hassle free? Use public transport (Taxi). Come on weekdays. U wanna get a good photos, come by night time. Need a help for a good photo spot, you may asked a kinda of tour guide inside. No parking space.....
3	5.0	Travelwith Arben	It is quite interesting to experience such a historic place in the city of Semarang, the place kinda scary but we both amazed by the story behind the building. I really recommend a visit while you are in town.
4	3.0	Stefanus Wijaya	This place previously known as haunted house, but now already renovated and become public space and museum by Indonesian Railways Company (KAI). It has exquisite exterior design, however inside this place not well maintaned and quite empty for museum. Hopefully there will be improvement in the future.
5	4.0	Ubud98	Good, one of Semarang city's landmark. Suggest to visit at night. Very close to Semarang Catholic Cathedral.

Tabel 4.2 Data Ulasan Berbahasa Indonesia

No.	Penilaian	Nama	Ulasan
1	5.0	Ignatius M	Rame dengan pengunjung, letaknya dekat kemana - mana. Disekitaran Lawang Sewu juga banyak pilihan hotel dan dekat Gereja Katedral
2	5.0	Deddy	Cocok untuk wisata edukasi sejarah. kalau mengunjungi tempat sejarah bersama anak2 sebaiknya di siang hari
3	5.0	Rizenda	Sangat bagus untuk pembelajaran dan wisata.. kurang leluasa dalam mengeksplor seluruh ruangan. Kurang tersedia tempat rehat setelah berkeliling.
4	5.0	Just Mira	Bagus kl datang pas malem, cakep bt foto2 cuma rada2 ngeri2 sedap ...angker karna bangunan kuno di dlm gedung banyak foto2 bersejarah tentang gedung dan alat2 telekomunikasi dari masa ke masa..
5	5.0	Kurniadi	Bagus Sekali. Saya berkunjung kamis malam (malam jumat) ke Lawang Sewu Semarang. Sengaja menggunakan jasa guide supaya mendapat penjelasan dan mendapat akses ke semua bagian gedung. Secara keseluruhan gedung ini masih bagus dan terawat. Pada malam hari tampak aroma 'seram' tetapi tetap terlihat 'gagah'. Gedung bersejarah, arsitektural indah dan terpelihara baik. Recommended dech

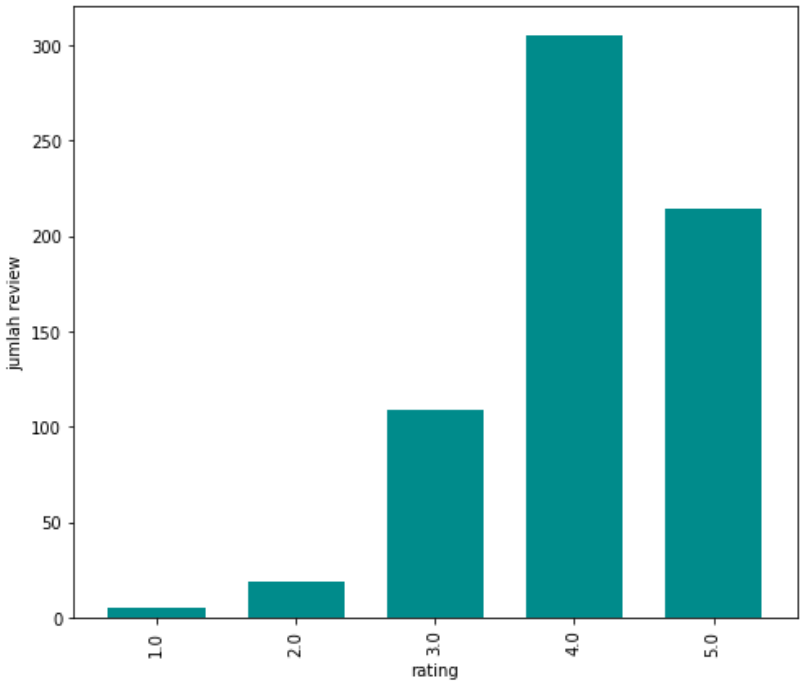
4.1.2 Analisis Deskriptif

Analisis deskriptif dalam penelitian ini berfungsi untuk mendeskripsikan, atau memberi gambaran secara umum tentang pemberian penilaian/rating oleh para pengunjung objek wisata Lawang Sewu berdasarkan pada data ulasan dari situs TripAdvisor. Tujuannya untuk mengetahui berapa banyak ulasan pada tiap penilaian/rating. Data ulasan berbahasa Inggris dapat diketahui analisis deskriptif rating pengunjung dengan *script* seperti berikut.

```
#melihat banyak ulasan dari tiap rating
df['rating'].value_counts()
```

Setiap rating memiliki arti yang berbeda, pada situs TripAdvisor rating 1 sampai 5 secara berurutan memiliki arti "*terrible*", "*poor*", "*average*", "*very good*" dan "*excellent*". Berdasar *output script* Python diatas dapat diketahui bahwa sebagian besar ulasan pengunjung Lawang Sewu berbahasa Inggris memberikan rating baik, dengan rincian sebagai berikut, dari 652 ulasan, tercatat rating *excellent* sebanyak 214 ulasan, rating *very good* sebanyak 305 ulasan, rating *average* sebanyak 109 ulasan, rating *poor* sebanyak 19 ulasan, dan rating *terrible* sebanyak 5 ulasan. Untuk melihat distribusi banyaknya ulasan per rating, dibuat diagram dengan *script* sebagai berikut, dan diagram distribusi rating tersebut dapat dilihat pada Gambar 4.1.

```
#melihat distribusi banyak ulasan berdasar rating
import matplotlib.pyplot as plt
_, ax1 = plt.subplots(figsize=(7,6))
stars_histogram = df['rating'].value_counts().sort_index()
stars_histogram.plot(kind='bar', width=0.7, color='darkcyan')
plt.xlabel('rating')
plt.ylabel('jumlah review')
plt.tight_layout()
plt.show()
```



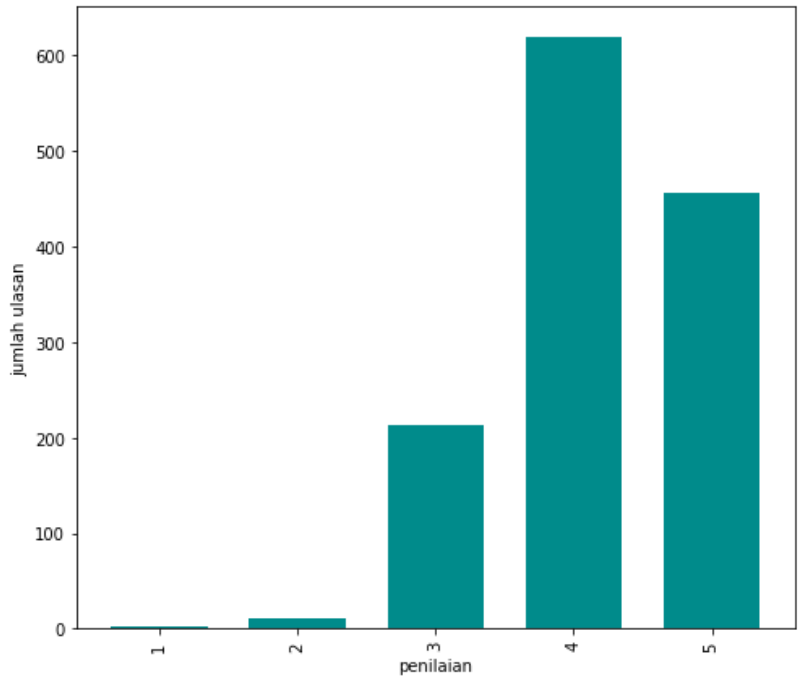
Gambar 4.1 Diagram *Rating* Data Ulasan Berbahasa Inggris

Data ulasan berbahasa Indonesia dapat diketahui analisis deskriptif penilaian pengunjung seperti berikut.

```
#melihat banyak ulasan dari tiap penilaian
df['penilaian'].value_counts()
```

Penilaian 1 samai 5 dalam ulasan TripAdvisor berbahasa Indonesia secara berurutan memiliki arti “sangat buruk”, “buruk”, “rata-rata”, “sangat bagus” dan “luar biasa”. Berdasar *output script* Python diatas dapat diketahui bahwa sebagian besar ulasan pengunjung Lawang Sewu berbahasa Indonesia juga memberikan penilaian baik, dengan rincian sebagai berikut, dari 1303 ulasan, tercatat penilaian luar biasa sebanyak 456 ulasan, penilaian sangat bagus sebanyak 620 ulasan, penilaian rata-rata sebanyak 213 ulasan, penilaian buruk sebanyak 11 ulasan, dan penilaian sangat buruk sebanyak 3 ulasan. Untuk melihat distribusi banyaknya ulasan per penilaian, dibuat diagram dengan *script* sebagai berikut, dan diagram distribusi penilaian tersebut dapat dilihat pada Gambar 4.2.

```
#melihat distribusi banyak ulasan berdasar penilaian
import matplotlib.pyplot as plt
_, ax1 = plt.subplots(figsize=(7,6))
stars_histogram = df['penilaian'].value_counts().sort_index()
stars_histogram.plot(kind='bar', width=0.7, color='darkcyan')
plt.xlabel('penilaian')
plt.ylabel('jumlah ulasan')
plt.tight_layout()
plt.show()
```



Gambar 4.2 Diagram Penilaian Data Ulasan Berbahasa Indonesia

4.1.3 Text Preprocessing

Data yang diambil dari situs TripAdvisor dengan cara *scrap* adalah data teks berupa *file* csv yang tidak terstruktur (*unstructured*). Oleh karena itu, diperlukan proses pengubahan bentuk menjadi data yang terstruktur. Data teks perlu dibersihkan sebelum masuk ke model *machine learning*, proses pembersihan ini disebut dengan *text preprocessing*. Terdapat beberapa tahapan dalam proses *text preprocessing* antara lain yaitu *cleansing*, *tokenizing*, *filtering* dan lain-lain. Penerapan tahap *text preprocessing* dapat berbeda-beda untuk setiap bahasa tergantung dengan kebutuhan,

karena perbedaan dalam bahasa tentu memiliki arti yang berbeda untuk tiap katanya. Tahap *text preprocessing* secara lengkapnya akan dibahas disub-sub bab tersendiri seperti berikut.

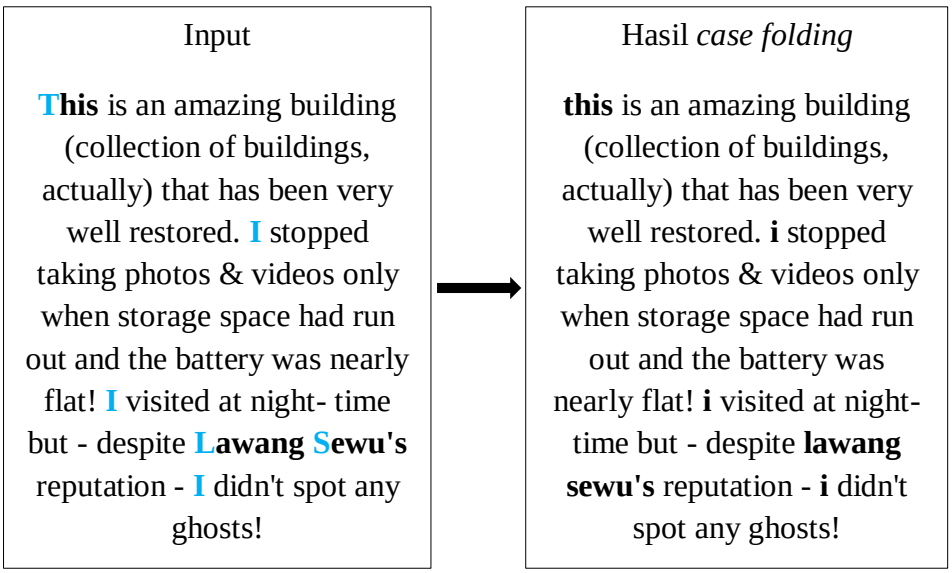
4.1.3.1 Text Preprocessing Data Berbahasa Inggris

Proses *text preprocessing* pada data ulasan berbahasa Inggris ada beberapa tahap yaitu, *case folding*, *filtering*, *cleansing*, *remove number*, *remove punctuation*, *remove short word*, *tokenizing* dan *remove extend stopword*.

4.1.3.1.1 Case Folding

Case folding adalah tahapan *text preprocessing* yang bertujuan untuk mengubah semua huruf dalam dokumen menjadi huruf kecil (*lower text*). *Script* untuk proses *case folding* adalah sebagai berikut, dan contoh penerapan *case folding* pada data ulasan berbahasa Inggris dapat dilihat pada Gambar 4.3.

```
#mengubah semua huruf dalam dokumen menjadi huruf kecil (lower
text)
df['review'] = df['review'].str.lower()
df.head()
```

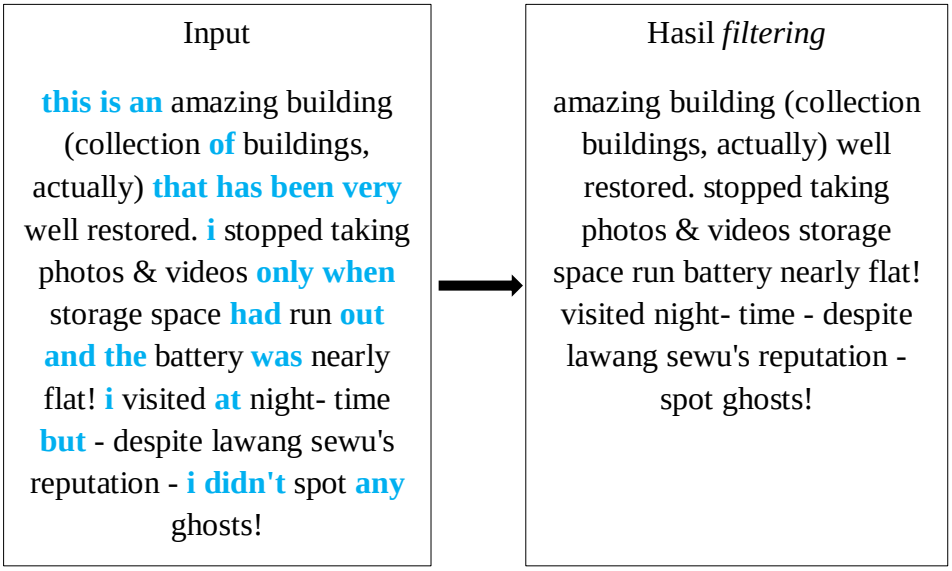


Gambar 4.3 Proses *Case Folding* Data Berbahasa Inggris

4.1.3.1.2 Filtering

Filtering adalah tahapan *text preprocessing* yang bertujuan untuk pembersihan teks dari *stopword*. *Stopword* adalah kata-kata yang tidak memiliki makna. Contoh *stopword* dalam bahasa Inggris adalah *"the", "and", "but", "if", "or"*. *Script* untuk proses *filtering* adalah sebagai berikut, dan contoh penerapan *filtering* pada data ulasan berbahasa Inggris dapat dilihat pada Gambar 4.4.

```
#menghapus stopword
from nltk.corpus import stopwords
stop_words = stopwords.words('english')
#fungsi untuk menghapus stopword
def remove_stopwords(rev):
    rev_new = " ".join([i for i in rev if i not in stop_words])
    return rev_new
df.head()
#menghapus stopword dari teks
df['review'] = [remove_stopwords(r.split()) for r in df['review
]]
df.head()
```

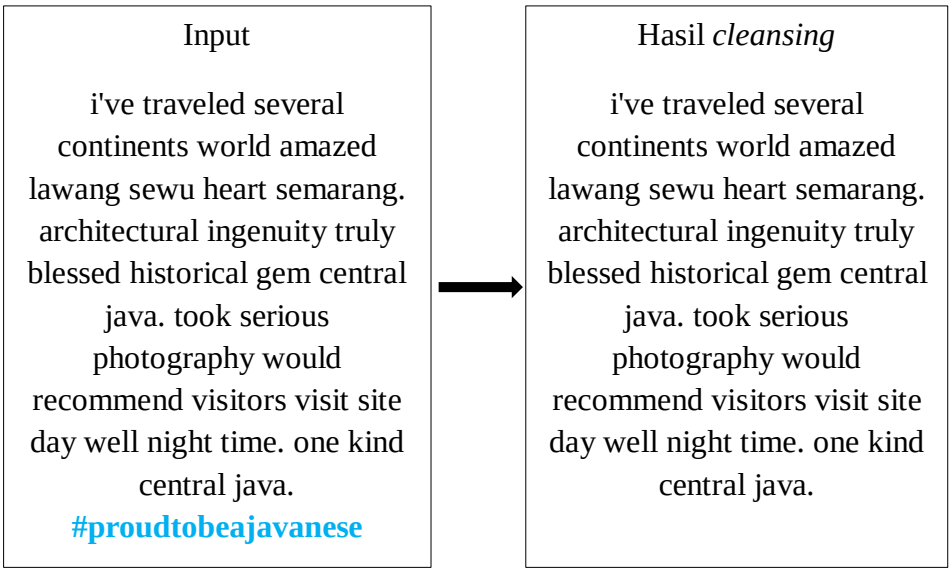


Gambar 4.4 Proses *Filtering* Data Berbahasa Inggris

4.1.3.1.3 Cleansing

Cleansing adalah tahapan *text preprocessing* yang bertujuan untuk pembersihan teks dari *tab, new line, back slice, mention, link, hastag, URL*. *Script* untuk proses *cleansing* adalah sebagai berikut, dan contoh penerapan *cleansing* pada data ulasan berbahasa Inggris dapat dilihat pada Gambar 4.5.

```
#pembersihan teks dari tab, new line, back slice, mention, link,
hastag, URL
import string
import re #regex library
def remove_ulasan_special(text):
    #menghapus tab, new line, dan back slice
    text = text.replace('\\t'," ").replace('\\n'," ").replace
    ('\\u'," ").replace('\\',"")
    #menghapus non ASCII (emoticon, chinese word, .etc)
    text = text.encode('ascii', 'replace').decode('ascii')
    #menghapus mention, link, hashtag
    text = ' '.join(re.sub("([@#][A-Zaz09]+)|(\w+:\/\/\S+)"," ",
    text).split())
    #menghapus incomplete URL
    return text.replace("http://", " ").replace("https://", " ")
df['review'] = df['review'].apply(remove_ulasan_special)
df.head()
```

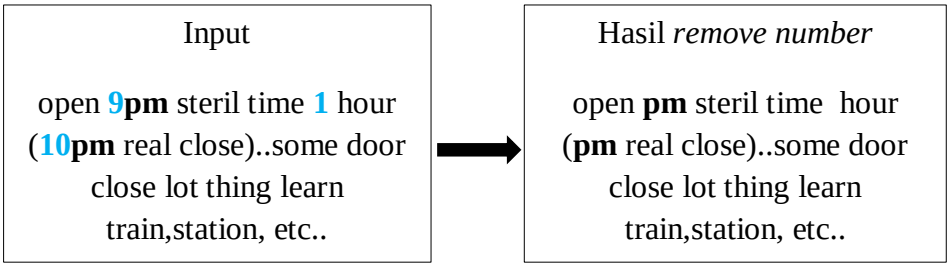


Gambar 4.5 Proses *Cleansing* Data Berbahasa Inggris

4.1.3.1.4 Remove Number

Remove number adalah tahapan *text preprocessing* yang bertujuan untuk pembersihan teks dari angka. *Script* untuk proses *remove number* adalah sebagai berikut, dan contoh penerapan *remove numer* pada data ulasan berbahasa Inggris dapat dilihat pada Gambar 4.6.

```
#menghapus angka
def remove_number(text):
    return  re.sub(r"\d+", "", text)
df['review'] = df['review'].apply(remove_number)
df.head()
```

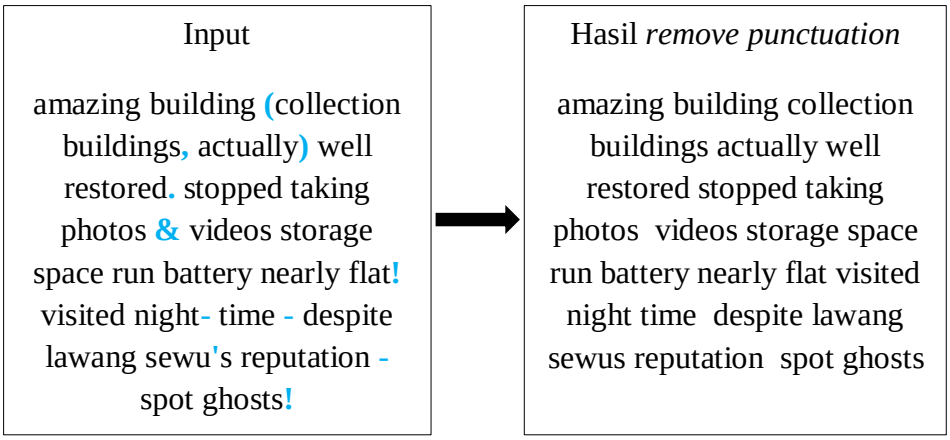


Gambar 4.6 Proses *Remove Number* Data Berbahasa Inggris

4.1.3.1.5 Remove Punctuation

Remove punctuation adalah tahapan *text preprocessing* yang bertujuan untuk pembersihan teks dari tanda baca. *Script* untuk proses *remove punctuation* adalah sebagai berikut, dan contoh penerapan *remove punctuation* pada data ulasan berbahasa Inggris dapat dilihat pada Gambar 4.7.

```
#menghapus tanda baca
def remove_punctuation(text):
    return text.translate(str.maketrans("","",string.punctuation
    ))
df['review'] = df['review'].apply(remove_punctuation)
df.head()
```

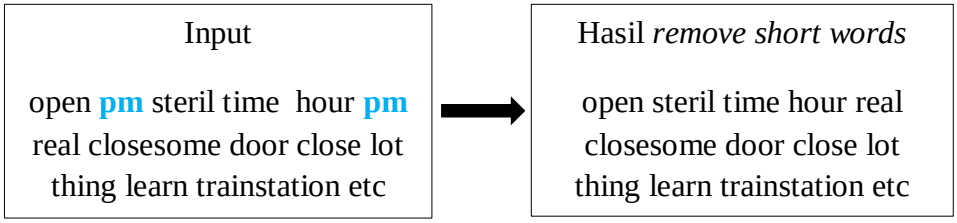


Gambar 4.7 Proses *Remove Punctuation* Data Berbahasa Inggris

4.1.3.1.6 Remove Short Words.

Remove short words adalah tahapan *text preprocessing* yang bertujuan untuk pembersihan teks dari kata-kata pendek (kata memiliki huruf kurang dari 3 huruf) yang dianggap tidak penting. Contoh kata adalah *"pm", "ex", "ur", "go". Script* untuk proses *remove short words* sebagai berikut, dan contoh penerapan *remove short words* pada data ulasan berbahasa Inggris dapat dilihat pada Gambar 4.8.

```
#menghapus kata-kata pendek (kurang dari 3 huruf)
df['review'] = df['review'].apply(lambda x: ' '.join([w for w in
x.split() if len(w)>2]))
df.head()
```

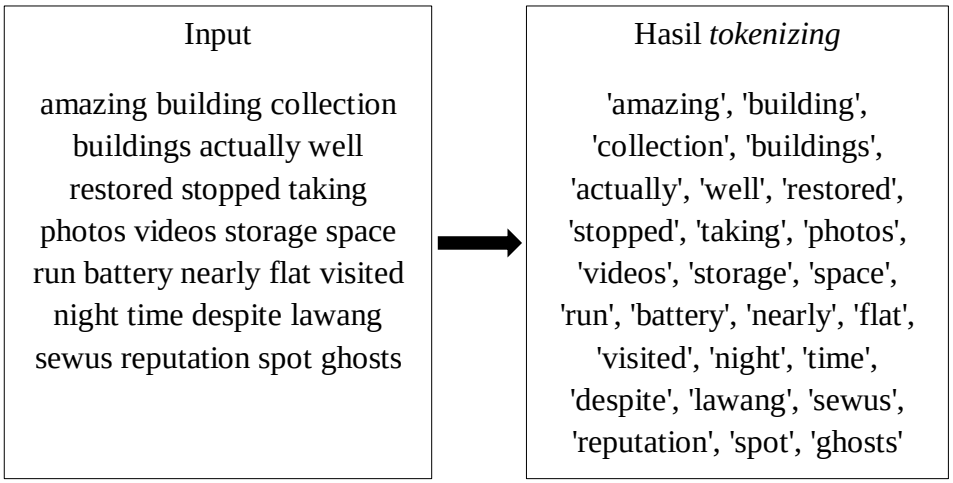


Gambar 4.8 Proses *Remove Short Words* Data Berbahasa Inggris

4.1.3.1.7 Tokenizing

Tokenizing adalah tahapan *text preprocessing* yang bertujuan untuk pemisahan teks menjadi potongan-potongan kata yang disebut sebagai token. Tujuan *tokenizing* agar data dapat diproses pada tahap selanjutnya yaitu *remove extend stopword*. *Script* untuk proses *tokenizing* adalah sebagai berikut, dan contoh penerapan *tokenizing* pada data ulasan berbahasa Inggris dapat dilihat pada Gambar 4.9.

```
#pemisahan teks menjadi potongan-potongan kata yang disebut
sebagai token
from nltk.tokenize import word_tokenize
#NLTK word tokenize
def word_tokenize_wrapper(text):
    return word_tokenize(text)
df['review'] = df['review'].apply(word_tokenize_wrapper)
df.head()
```



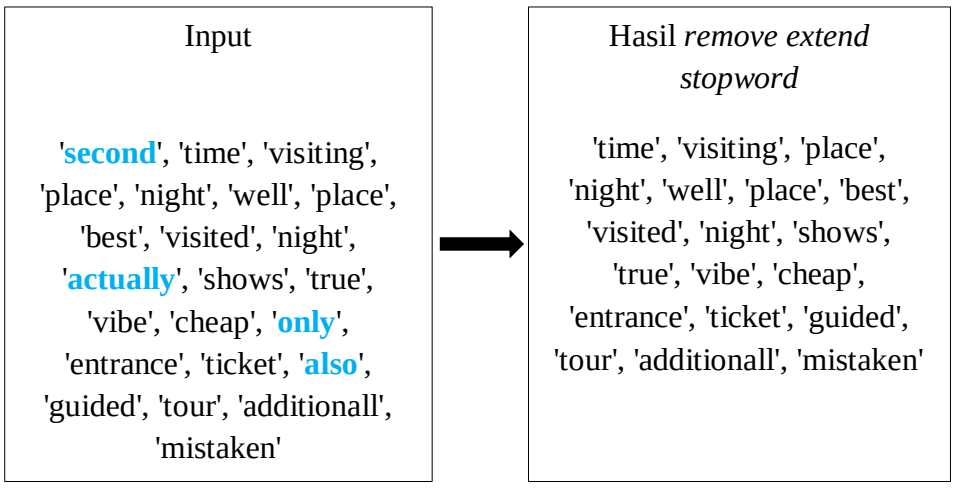
Gambar 4.9 Proses *Tokenizing* Data Berbahasa Inggris

4.1.3.1.8 Remove Extend Stopword

Remove extend stopword adalah tahapan *text preprocessing* yang bertujuan untuk pembersihan teks dari *stopword* yang sebelumnya tidak terhapus pada proses *filtering*. Pada proses *filtering* untuk menghapus *stopword* digunakan *library* NLTK, dimana dalam *library* tersebut hanya terdapat 179 kata *stopword*. *Stopword* tambahan dapat ditambahkan secara manual perkata kedalam list *stopword* dengan

menggunakan fungsi .extend(), penambahan *stopword* juga dapat dibuat terlebih dahulu kamus *stopword* dalam bentuk *file* txt. Contoh *stopword* tambahan dalam bahasa Inggris antara lain *"thing", "also", "many", "even", "along". Script* untuk proses *remove extend stopword* adalah sebagai berikut, dan contoh penerapan *remove extend stopword* pada data ulasan berbahasa Inggris dapat dilihat pada Gambar 4.10.

```
#memfilter untuk mengambil kata-kata penting dengan menggunakan
algoritma stoplist (membuang kata kurang penting)
#import stopwords from NLTK
from nltk.corpus import stopwords
#dapatkan stopword dari NLTK stopword
stop_words = stopwords.words('english')
#menambahkan stopword tambahan
stop_words.extend(['thats', 'itll', 'youll'])
#menambahkan stopword dari file csv
txt_stopword = pd.read_csv('C:/Users/ACER/Tugas Akhir/stopword
inggris.txt', names= ["stopwords"], header = None)
#mengkonversi kata stopword tambahan & stopword dari file csv ke
list
stop_words.extend(txt_stopword["stopwords"][0].split(' '))
#mengkonversi list ke dictionary
stop_words = set(stop_words)
#menghapus stopword pada data
def stopwords_removal(words):
    return [word for word in words if word not in stop_words]
df['review'] = df['review'].apply(stopwords_removal)
df.head()
```



Gambar 4.10 Proses *Remove extend Stopword* Data Berbahasa Inggris

4.1.3.2 Text Preprocessing Data Berbahasa Indonesia

Proses *text preprocessing* pada data ulasan berbahasa Indonesia terdapat beberapa tahap yaitu, *case folding, cleansing, remove number, remove punctuation, remove single char, tokenizing, spell normalization* dan *filtering*.

4.1.3.2.1 Case Folding

Case folding adalah tahapan *text preprocessing* yang bertujuan untuk bertujuan untuk mengubah semua huruf dalam dokumen menjadi huruf kecil (*lower text*). *Script* untuk proses *case folding* adalah sebagai berikut, dan contoh penerapan *case folding* pada data ulasan berbahasa Indonesisa dapat dilihat pada Gambar 4.11.

```
#mengubah semua huruf dalam dokumen menjadi huruf kecil
(lower text)
df['ulasan'] = df['ulasan'].str.lower()
df.head()
```

Input	Hasil *case folding*
Rame dengan pengunjung, letaknya dekat kemana - mana. **D**isekitaran **L**awang **S**ewu juga banyak pilihan hotel dan dekat **G**ereja **K**atedral	**rame** dengan pengunjung, letaknya dekat kemana - mana. **disekitaran lawang sewu** juga banyak pilihan hotel dan dekat **gereja katedral**

Gambar 4.11 Proses *Case Folding* Data Berbahasa Indonesia

4.1.3.2.2 Cleansing

Cleansing adalah tahapan *text preprocessing* yang bertujuan untuk pembersihan teks dari *tab, new line, back slice, mention, link, hastag, URL*. *Script* untuk proses *cleansing* adalah sebagai berikut, dan contoh penerapan *cleansing* pada data ulasan berbahasa Indonesisa dapat dilihat pada Gambar 4.12.

```
#pembersihan teks dari tab, new line, back slice, mention, link,
hastag, URL
import string
import re #regex library
def remove_ulasan_special(text):
    #menghapus tab, new line, dan back slice
    text = text.replace('\\t'," ").replace('\\n'," ").replace
    ('\\u'," ").replace('\\',"")
    #menghapus non ASCII (emoticon, chinese word, .etc)
    text = text.encode('ascii', 'replace').decode('ascii')
    #menghapus mention, link, hashtag
    text = ' '.join(re.sub("([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)",
    " ",text).split())
    #menghapus incomplete URL
    return text.replace("http://", " ").replace("https://", " ")
df['ulasan'] = df['ulasan'].apply(remove_ulasan_special)
df.head()
```

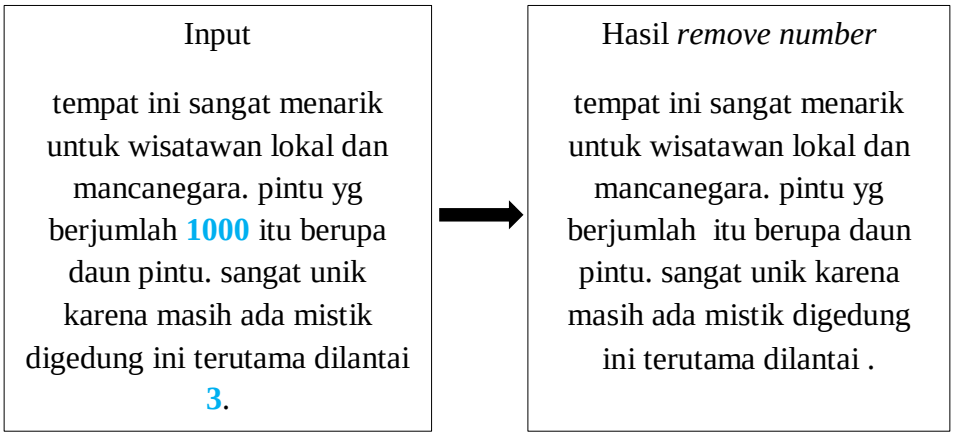
Input	Hasil *cleansing*
karena senang mengunjungi museum, maka lawang sewu adalah salah satu destinasi wajib bila berada di semarang. cobalah berkunjung di malam hari. rasakan sensasi adrenalin dobel dan menurut saya menjadi lebih terkesan mistis,, **#wisatamalam**	karena senang mengunjungi museum, maka lawang sewu adalah salah satu destinasi wajib bila berada di semarang. cobalah berkunjung di malam hari. rasakan sensasi adrenalin dobel dan menurut saya menjadi lebih terkesan mistis,,

Gambar 4.12 Proses *Cleansing* Data Berbahasa Indonesia

4.1.3.2.3 Remove Number

Remove number adalah tahapan *text preprocessing* yang bertujuan untuk pembersihan teks dari angka. *Script* untuk proses *remove number* adalah sebagai berikut, dan contoh penerapan *remove number* pada data ulasan berbahasa Indonesisa dapat dilihat pada Gambar 4.13.

```
# menghapus angka
def remove_number(text):
    return  re.sub(r"\d+", "", text)
df['ulasan'] = df['ulasan'].apply(remove_number)
df.head()
```

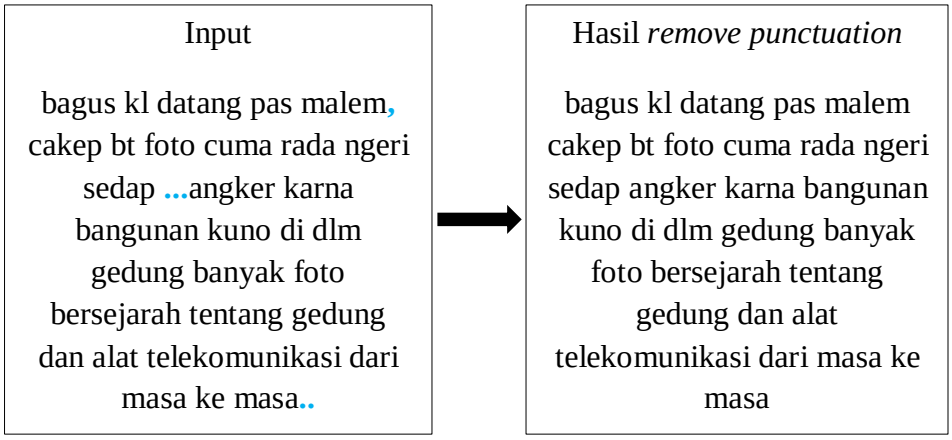


Gambar 4.13 Proses *Remove Number* Data Berbahasa Indonesia

4.1.3.2.4 Remove Punctuation

Remove punctuation adalah tahapan *text preprocessing* yang bertujuan untuk pembersihan teks dari tanda baca. *Script* untuk proses *remove punctuation* adalah sebagai berikut, dan contoh penerapan *remove punctuation* pada data ulasan berbahasa Indonesisa dapat dilihat pada Gambar 4.14.

```
#menghapus tanda baca
def remove_punctuation(text):
    return text.translate(str.maketrans("","",string.punctuation
    ))
df['ulasan'] = df['ulasan'].apply(remove_punctuation)
df.head()
```

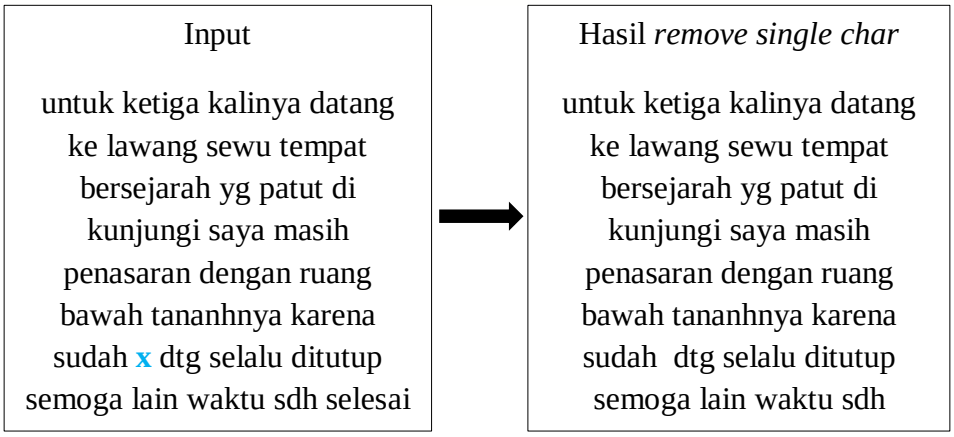


Gambar 4.14 Proses *Remove Punctuation* Data Berbahasa Indonesia

4.1.3.2.5 Remove Single Char

Remove single char adalah tahapan *text preprocessing* yang bertujuan untuk pembersihan teks dari huruf tunggal (tidak berbentuk kata). *Script* untuk proses *remove single char* adalah sebagai berikut, dan contoh penerapan *remove single char* pada data ulasan berbahasa Indonesia dapat dilihat pada Gambar 4.15.

```
#menghapus huruf tunggal (tidak berbentuk kata)
def remove_singl_char(text):
    return re.sub(r"\b[a-zA-Z]\b", "", text)
df['ulasan'] = df['ulasan'].apply(remove_singl_char)
df.head()
```



Gambar 4.15 Proses *Remove Single Char* Data Berbahasa Indonesia

4.1.3.2.6 Tokenizing

Tokenizing adalah tahapan *text preprocessing* yang bertujuan untuk pemisahan teks menjadi potongan-potongan kata yang disebut sebagai token. Tujuan *tokenizing* agar data dapat diproses pada tahap selanjutnya yaitu *spell normalization* dan *filtering*. *Script* untuk proses *tokenizing* adalah sebagai berikut, dan contoh penerapan *tokenizing* pada data ulasan berbahasa Indonesisa dapat dilihat pada Gambar 4.16.

```
#pemisahan teks menjadi potongan-potongan kata yang disebut
sebagai token
from nltk.tokenize import word_tokenize
#NLTK word tokenize
def word_tokenize_wrapper(text):
    return word_tokenize(text)
df['ulasan'] = df['ulasan'].apply(word_tokenize_wrapper)
df.head()
```

Input	Hasil *tokenizing*
untuk ketiga kalinya datang ke lawang sewu tempat bersejarah yg patut di kunjungi saya masih penasaran dengan ruang bawah tananhnya karena sudah dtg selalu ditutup semoga lain waktu sdh selesai renovasi	'untuk', 'ketiga', 'kalinya', 'datang', 'ke', 'lawang', 'sewu', 'tempat', 'bersejarah', 'yg', 'patut', 'di', 'kunjungi', 'saya', 'masih', 'penasaran', 'dengan', 'ruang', 'bawah', 'tananhnya', 'karena', 'sudah', 'dtg', 'selalu', 'ditutup', 'semoga', 'lain', 'waktu', 'sdh', 'selesai', 'renovasi'

Gambar 4.16 Proses *Tokenizing* Data Berbahasa Indonesia

4.1.3.2.7 Spell Normalization

Spell normalization adalah tahapan *text preprocessing* yang bertujuan untuk perbaikan kata-kata yang salah penulisannya (*typo*) atau kata-kata yang penulisannya disingkat. *Spell normalization* juga bertujuan untuk menyeragamkan kata yang

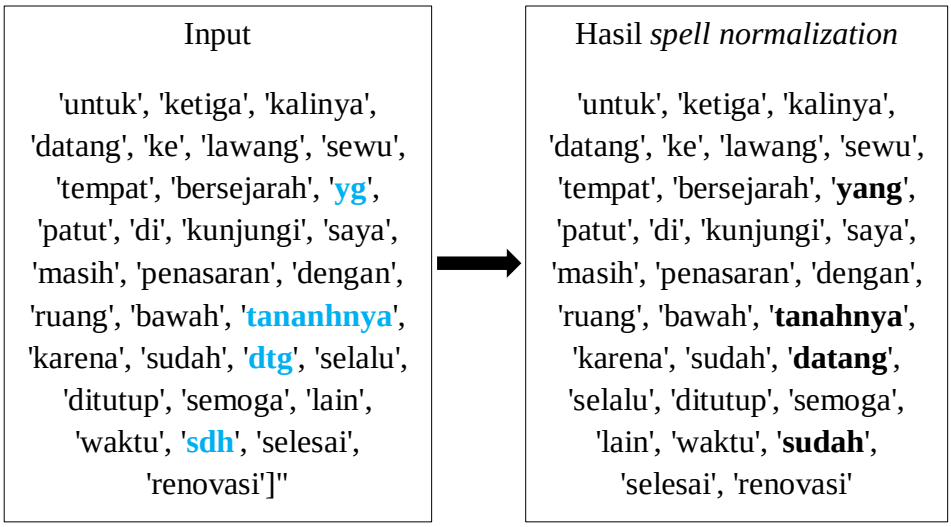
memiliki makna yang sama namun penulisannya berbeda. Contoh perbaikan kata seperti berikut.

baatss	: banget	karean	: karena
baget	: banget	karene	: karena
banged	: banget	karna	: karena
bangeed	: banget	krn	: karena
bangeeeettttttt	: banget	krna	: karena
bangetttt	: banget	kemareeen	: kemarin
bgt	: banget	kmren	: kemarin
bngt	: banget	kmrin	: kemarin
buangeeeedd	: banget	kmrn	: kemarin

Gambar 4.17 Contoh Perbaikan Kata pada Proses *Spell Normalization*

Script untuk proses *spell normalization* adalah sebagai berikut, dan contoh penerapannya dapat dilihat ada Gambar 4.18.

```
#menyeragamkan kata yang memiliki makna yang sama namun
penulisannya berbeda
normalizad_word = pd.read_excel("normalisasi.xlsx")
normalizad_word_dict = {}
for index, row in normalizad_word.iterrows():
    if row[0] not in normalizad_word_dict:
        normalizad_word_dict[row[0]] = row[1]
def normalized_term(document):
    return [normalizad_word_dict[term] if term in normalizad_wor
    d_dict else term for term in document]
df['ulasan'] = df['ulasan'].apply(normalized_term)
df.head()
```



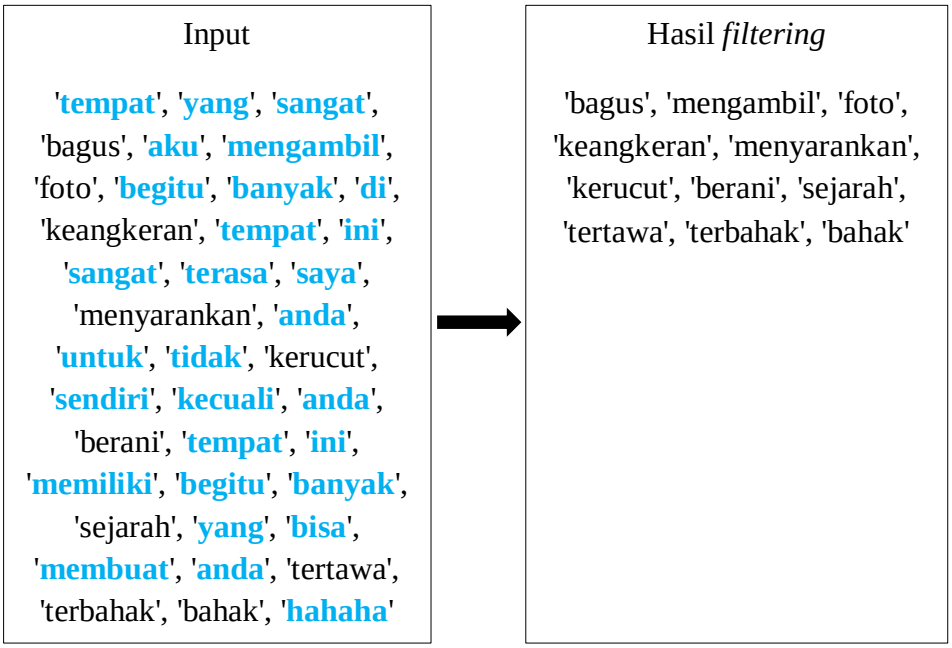
Gambar 4.18 Proses *Spell Normalization* Data Berbahasa Indonesia

4.1.3.2.8 Filtering

Filtering adalah tahapan *text preprocessing* yang bertujuan untuk pembersihan teks dari *stopword*. *Stopword* adalah kata-kata yang tidak memiliki makna. Pada proses *filtering* untuk menghapus *stopword* digunakan *library* NLTK, dimana dalam *library* tersebut hanya terdapat 758 kata *stopword*. *Stopword* tambahan dapat ditambahkan secara manual perkata ke dalam list *stopword* dengan menggunakan fungsi .extend(), penambahan *stopword* juga dapat dibuat terlebih dahulu kamus *stopword* dalam *file* txt. Contoh *stopword* dalam bahasa Indonesia adalah “yang”, “untuk”, “dan”, “pada”, “dengan”. *Script* untuk proses *filtering* adalah sebagai berikut, dan contoh penerapan filtering pada data ulasan berbahasa Indonesisa dapat dilihat pada Gambar 4.19.

```
#memfilter untuk mengambil kata-kata penting dengan menggunakan
algoritma stoplist (membuang kata kurang penting)
#import stopword from NLTK
from nltk.corpus import stopwords
#dapatkan stopword indonesia dari NLTK stopword
list_stopwords = stopwords.words('indonesian')
#menambahkan stopword tambahan
list_stopwords.extend(['pas', 'ya', 'sih', 'deh', 'loh', 'oiya',
'nih', 'ok', 'ah'])
#meambahkan stopword dari file csv
```

```
txt_stopword = pd.read_csv('C:/Users/ACER/Tugas Akhir/stopword
indonesia.txt', names= ["stopwords"], header = None)
#mengkonversi kata stopword tambahan & stopword pada file csv
ke list
list_stopwords.extend(txt_stopword["stopwords"][0].split(' '))
#mengkonversi list ke dictionary
list_stopwords = set(list_stopwords)
#menghapus stopword pada data
def stopwords_removal(words):
    return [word for word in words if word not in list_stopwords
    ]
df['ulasan'] = df['ulasan'].apply(stopwords_removal)
df.head()
```



Gambar 4.19 Proses *Filtering* Data Berbahasa Indonesia

4.1.4 Pembobotan TF-IDF

Data ulasan yang sudah melewati tahap *text preprocessing* masih berbentuk teks (kata), dalam analisis klasifikasi data harus numerik atau angka. Data harus dikonversi terlebih dahulu ke bentuk angka dengan menggunakan pembobotan kata dengan TF-IDF. Proses pembobotan kata dengan TF-IDF dilakukan dengan *script*

sebagai berikut dan menampilkan *term* yang memiliki bobot terbesar, hasil *output script* ditampilkan pada Tabel 4.3.

```
#pembobotan kata
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
cvec = CountVectorizer(stop_words='english', min_df=1, max_df=.5
, ngram_range=(1,2))
cvec
#Bobot untuk setiap istilah dalam setiap dokumen
transformer = TfidfTransformer()
transformed_weights = transformer.fit_transform(cvec_counts)
transformed_weights
#melihat term teratas dengan weight rata-rata tf-idf
weights = np.asarray(transformed_weights.mean(axis=0)).ravel().
tolist()
weights_df = pd.DataFrame({'term': cvec.get_feature_names(),
'weight': weights})
weights_df.sort_values(by='weight', ascending=False).head(20)
```

Tabel 4.3 Pembobotan Kata TF-IDF

Bahasa Inggris		**Bahasa Indonesia**	
Term	**Weight**	**Term**	**Weight**
Place	0,050128	Bangunan	0,055212
Visit	0,038192	Gedung	0,053636
History	0,032755	Sejarah	0,038385
Doors	0,031293	Pintu	0,033533
Guide	0,030098	Kota	0,032356
Good	0,026942	Bagus	0,031646
old	0,025488	Foto	0,031478

4.1.5 Analisis Sentimen

Setelah dilakukan pembobotan kata, selanjutnya akan dilakukan pelabelan kelas sentimen. Menurut sumber datanya analisis sentimen dibedakan menjadi dua jenis, yaitu analisis pada level dokumen dan analisis pada level kalimat. Dalam penelitian ini data akan dilakukan analisis sentimen untuk tiap ulasan, maka dalam

penelitian ini masuk ke dalam jenis analisis sentimen pada level kalimat (*fined grained sentiment analysis*).

Ada tiga cara pelabelan dalam analisis sentimen ulasan, yaitu berdasarkan nilai rating, berdasarkan kamus lexicon dan pemberian label secara manual dengan membaca ulasan satu-persatu. Akan dijelaskan kelebihan dan kekurangan dari masing-masing cara pelabelan.

1. Pelabelan berdasarkan nilai rating: pelabelan ini biasanya melabelkan data menjadi kelas sentimen positif dan sentimen negatif berdasar nilai rating yang telah ditentukan, misalnya data akan bersentimen positif jika nilai rating lima (rating = 5) dan akan bersentimen negatif jika nilai rating selain lima atau kurang dari lima (rating < 5). Kelebihan dari cara ini adalah mudah diterapkan dan digunakan, hanya perlu merubah nilai rating dengan rumus if. Tetapi kelemahan dari cara ini adalah bahwa nilai rating tidak selalu menunjukan bahwa ulasan dengan rating yang baik akan bersentimen positif, misalnya dalam ulasan yang mengandung kalimat negasi.
2. Pelabelan berdasar kamus lexicon: pelabelan dengan cara ini melabelkan data dengan memilah satu ulasan berdasar kata kedalam kata positif atau negatif, jika kata tidak terdapat pada kata positif dan negatif, maka kata akan terhitung sebagai kata netral. Kemudian jumlah kata tersebut akan dikalkulasi dan dapat diketahui ulasan tersebut masuk ke dalam sentimen positif, netral atau negatif. Kelebihan dari cara ini adalah lebih akurat jika dibandingkan dengan cara rating, karena cara ini akan memeriksa ulasan berdasar kata yang dikandungnya. Namun kelemahan dari cara ini adalah jika terdapat kata dalam kamus lexicon yang tidak sesuai dengan keadaan pada kenyataannya.
3. Pelabelan manual dengan membaca ulasan: pelabelan dengan cara ini dilakukan dengan membaca ulasan satu persatu, lalu akan ditentukan ulasan tersebut masuk kedalam sentimen positif atau negatif. Kelebihan dari pelabelan dengan cara ini adalah hasil yang didapatkan akan lebih sesuai dengan kenyataan dan tidak memperlukan proses komputasi. Kelemahan dari cara ini adalah dikhawatirkan

adanya kecenderungan opini seseorang dan cara ini akan sulit diterapkan pada data yang berjumlah besar.

Proses pelabelan kelas sentimen pada data ulasan berbahasa Inggris dilakukan secara otomatis menggunakan *library* vader lexicon untuk memanggil fungsi SentimentIntensityAnalyzer dari *package* NLTK. Data hanya akan dilabelkan kedalam dua kelas sentimen, yaitu sentimen positif dan sentimen negatif. Data diproses dengan menerapkan algoritma *Sentiment Analysis* Vader. Vader akan menghitung skor dari tiap kalimat (ulasan). Skor yang dihasilkan berupa positif, negatif dan netral. Setiap skor yang dihasilkan akan digabungkan sehingga menghasilkan nilai compound. Compound merupakan matriks yang menghitung semua skor yang telah dinormalisasi antara -1 dan +1. Nilai compound dibawah 0 (compound < 0) dinyatakan sebagai kelas negatif, sedangkan nilai compound diatas 0 (compound >= 0) dinyatakan sebagai kelas positif. Atau dapat juga ditentukan dari besarnya presentase kata positif dan negatif dalam satu kalimat tersebut. Jika dalam kalimat tersebut presentase kata positif lebih besar dari presentase kata negatif maka dinyatakan sebagai kelas positif dan begitu juga sebaliknya, jika dalam kalimat tersebut presentase kata negatif lebih besar dari presentase kata positif maka dinyatakan sebagai kelas negatif. *Script* untuk proses pelabelan data sebagai berikut, dan contoh hasil *output* untuk *script* pelabelan dapat dilihat pada Tabel 4.4.

```
#pelabelan data, data akan berlabel negatif jika nilai compound
<0,0 dan akan berlabel positif jika nilai compound >=0,0
import nltk
from nltk.sentiment.vader import SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()
df['sentiment'] = df['review'].apply(lambda x: sid.polarity_scor
es(x))
def convert(x):
    if x < 0.0:
        return "negative"
    #elif x > 0.0:
        #return "positive"
    else:
        return "positive"
df['result'] = df['sentiment'].apply(lambda x:convert(x['compoun
d']))
```

```
#df.groupby(['brand','result']).size()
#df.groupby(['brand','result']).count()
x = df.groupby(['review','name'])['result'].value_counts(normali
ze=True)
x = df.groupby(['review'])['result'].value_counts(normalize=True
)
y = x.loc[(x.index.get_level_values(1) == 'negative')]
#print(y[y>0.2])
print(df.shape)
df.head()
```

Tabel 4.4 Hasil Pelabelan Analisis Sentimen Data Berbahasa Inggris

Data	Sentimen	Hasil
amazing building collection buildings well restored stopped taking photos videos storage space run battery nearly flat visited night time despite reputation spot ghosts	Negative: 0.065, Neutral: 0.651, Positive: 0.284, Compound: 0.7506	Positive
interesting experience historic place place kinda scary amazed story behind building recommend visit	Negative: 0.145, Neutral: 0.449, Positive: 0.406, Compound: 0.6387	Positive
built literary doors oldest building formerly used administratiegebouw nederlandsindische railway company main tourist destination visit entrance fee adults explore building long want forget climb floor building find large ballroom right beneath roof	Negative: 0.057, Neutral: 0.904, Positive: 0.039, Compound: -0.1531	Negative
building scariest place people name meansthousand door another reason historical story place example building	Negative: 0.187, Neutral: 0.755, Positive: 0.058,	Negative

main office railway company pioneer train centuryand building change headquaterwhen invaded island creeepy story building started use building interogation room tortured room especially people fight independence tortured room roof top basement floor visit basement floor recomend night need flashlight boot rent item guide basement floor water nearest river build create cooler good air circulation carefull walk basement flloor dark guide tell creepy story prisoner die place throw dead body beside creepy basement floor visit train used island small museum train compant place visit	Compound: -0,9515	

Proses pelabelan kelas sentimen pada data ulasan berbahasa Indonesia, dilakukan secara otomatis dengan coding pelabelan yang berdasarkan kamus lexicon. untuk pelabelan data ulasan berbahasa Indonesia ini sedikit berbeda dengan data ulasan berbahasa Inggris sebelumnya, untuk pelabelan data berbahasa Inggris telah tersedia *package* dan fungsi tersendiri pada Python, sedangkan untuk data berbahasa Indonesia belum tersedia, karena perbedaan bahasa maka harus membuat sendiri *script* coding pelabelan tersebut. Konsep pelabelan pada data berbahasa Indonesia sama dengan pelabelan data berbahasa Inggris, disini pelabelan sama-sama dibagi menjadi dua kelas sentimen yaitu kelas sentimen positif dan kelas sentimen negatif . Nilai yang dihasilkan berupa nilai sentimen, nilai sentimen dibawah 0 (sentimen < 0) dinyatakan sebagai kelas negatif, sedangkan nilai sentimen diatas 0 (sentimen >= 0) dinyatakan sebagai kelas positif.

Tabel 4.5 Hasil Pelabelan Analisis Sentimen Data Berbahasa Indonesia

Data	Sentimen	Hasil
cocok wisata edukasi sejarah mengunjungi sejarah anak siang	1	Positif
bagus pembelajaran wisata leluasa mengeksplor ruang tersedia rehat berkeliling	4	Positif
lumayan ngeluarin kocek yak mitosnya percaya anjing	-1	Negatif
kunjungan teman berkunjung kesemua penjara tanahnya terawat parkiran mobil susah jangkau ojek online	-4	Negatif

Setelah proses pelabelan kelas dengan analisis sentimen, dapat diketahui jumlah ulasan yang berlabel positif dan negatif dengan *script* berikut.

```
#banyak data yang berlabel positif & negatif
df['result'].value_counts()
```

berdasarkan *output script* tersebut dapat diketahui pengunjung lebih banyak memberikan ulasan positif. Untuk data ulasan berbahasa Inggris diperoleh 555 ulasan masuk ke dalam kelas sentimen positif dan 97 ulasan masuk ke dalam kelas sentimen negatif, sedangkan untuk data ulasan berbahasa Indonesia diperoleh 1017 ulasan masuk ke dalam kelas sentimen positif dan 286 ulasan masuk ke dalam kelas sentimen negatif. *Output* tersebut diringkas dalam Tabel 4.6.

Tabel 4.6 Perbandingan Jumlah Data Ulasan pada Kelas Sentimen

Sentimen	Bahasa Inggris	Bahasa Indonesia
Positif	555	1017
Negatif	97	286

4.1.6 Data Latih dan Data Uji

Sebelum melanjutkan ke analisis klasifikasi, data ulasan perlu dibagi menjadi data latih dan data uji terlebih dahulu. Data latih digunakan untuk membentuk model klasifiksi (melatih) yang berisi pengetahuan yang nantinya akan digunakan untuk prediksi kelas sentimen yang baru. Semakin banyak data yang dilatih maka akan semakin baik algoritma tersebut dalam memahami data.

Data uji digunakan untuk melihat presentase algoritma klasifikasi berhasil melakukan klasifikasi dengan benar. Dalam analisis ini dibagi banyaknya data latih adalah 80% dan data uji 20% untuk masing-masing data. Untuk proses partisi data menjadi data latih dan data uji, digunakan *script* sebagai berikut dan hasil *output* ditampilkasn dalam Tabel 4.7.

```
#data training dan data testing
Train_X, Test_X, Train_Y, Test_Y = model_selection.train_test_
split(klasifikasiLawangSewu['ulasan'],klasifikasiLawangSewu
['label'],test_size=0.2, random_state=8)
Encoder = LabelEncoder()
Train_Y = Encoder.fit_transform(Train_Y)
Test_Y = Encoder.fit_transform(Test_Y)
Tfidf_vect = TfidfVectorizer(max_features=5000)
Tfidf_vect.fit(klasifikasiLawangSewu['ulasan'])
Train_X_Tfidf = Tfidf_vect.transform(Train_X)
Test_X_Tfidf = Tfidf_vect.transform(Test_X)
len(Train_X)
len(Test_X)
```

Tabel 4.7 Perbandingan Jumlah Data Latih dan Data Uji

Data	Bahasa Inggris	Bahasa Indonesia
Data Latih	521	1042
Data Uji	131	261

4.1.7 Klasifikasi

Prediksi klasifikasi dilakukan data uji dengan mempelajari pengetahuan-pengetahuan yang terdapat pada data latih. Dalam data latih terdapat kelas sentimen positif dan negatif yang akan dipelajari ciri-ciri kata-kata yang terdapat pada masing-

masing kelas sentimen. Dari lima metode klasifikasi didapat nilai akurasi, dengan *script* untuk masing-masing metode klasifikasi adalah sebagai berikut, dengan perbandingan nilai akurasinya diringkas dalam Tabel 4.8.

```
#Support Vector Mechine
SVM = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='scale')
SVM.fit(Train_X_Tfidf,Train_Y)
predictions_SVM = SVM.predict(Test_X_Tfidf)
print("SVM Accuracy Score -> ",accuracy_score(predictions_SVM,
Test_Y)*100)

#Naive Bayes
from sklearn import naive_bayes
NB = naive_bayes.MultinomialNB()
NB.fit(Train_X_Tfidf,Train_Y)
predictions_NB = NB.predict(Test_X_Tfidf)
print("NB Accuracy Score -> ",accuracy_score(predictions_NB,
Test_Y)*100)

#Decision Tree
from sklearn.tree import DecisionTreeClassifier
DT = DecisionTreeClassifier()
DT.fit(Train_X_Tfidf,Train_Y)
predictions_DT = DT.predict(Test_X_Tfidf)
print("DT Accuracy Score -> ",accuracy_score(predictions_DT,
Test_Y)*100)

#Random Forest
from sklearn.ensemble import RandomForestClassifier
RF = RandomForestClassifier()
RF.fit(Train_X_Tfidf,Train_Y)
predictions_RF = RF.predict(Test_X_Tfidf)
print("RF Accuracy Score -> ",accuracy_score(predictions_RF,
Test_Y)*100)

#Logistic Regression
from sklearn.linear_model import LogisticRegression
LR = LogisticRegression()
LR.fit(Train_X_Tfidf,Train_Y)
predictions_LR = LR.predict(Test_X_Tfidf)
print("LR Accuracy Score -> ",accuracy_score(predictions_LR,
Test_Y)*100)
```

Tabel 4.8 Perbandingan Akurasi Algoritma Klasifikasi

Metode	Bahasa Inggris	Bahasa Indonesia
Support Vector Mechine	89,3%	86,2%
Naïve Bayes	87,8%	78,9%
Decision Tree	76,3%	72,4%
Random Forest	87,8%	80,0%
Logistic Regression	87,8%	80,5%

Berdasar Tabel 4.8 didapatkan nilai akurasi dari masing-masing metode klasifikasi, dari nilai akurasi tersebut, baik pada hasil data berbahasa Inggris maupun data berbahasa Indonesia diketahui bahwa nilai akurasi untuk metode klasifikasi *Support Vector Mechine* adalah yang tertinggi. Nilai tersebut menunjukkan bahwa metode klasifikasi *Support Vector Mechine* paling baik untuk mengklasifikasikan data ulasan pengunjung Lawang Sewu.

Evaluasi model klasifikasi selain dilihat dari nilai akurasi dapat dilihat juga dari hasil *confusion matrix* untuk hasil klasifikasi data. *Confusion matrix* ini digunakan untuk mengevaluasi hasil dari prediksi dari metode klasifikasi, *script* untuk *confusion matrix* seperti berikut dan dihasilkan *output confusion matrix* seperti ada Tabel 4.9.

```
#confusion matrix
from sklearn.metrics import confusion_matrix
import seaborn as sns
conf_mat = confusion_matrix(Test_Y, predictions_SVM)
class_label = ["negative", "positive"]
test = pd.DataFrame(conf_mat, index = class_label, columns =
class_label)
sns.heatmap(test, annot = True,fmt="d")
plt.title("Confusion Matrix for test data")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.show()
```

Tabel 4.9 Hasil *Confusion Matrix*

Data	**Aktual**	**Prediksi**	
		Negatif	**Positif**
Bahasa Inggris	**Negatif**	2	14
	Positif	0	115
	Akurasi	**89,3%**	
Bahasa Indonesia	**Negatif**	23	32
	Positif	4	202
	Akurasi	**86,2%**	

Berdasar Tabel 4.9 pada data ulasan berbahasa Inggris dapat diketahui bahwa, terdapat 115 data positif yang benar terprediksi masuk kedalam kelas sentimen positif dan tidak ada data positif yang terprediksi masuk kedalam kelas sentimen negatif, serta 2 data negatif yang benar terprediksi masuk kedalam kelas sentimen negatif dan 14 data negatif yang terprediksi masuk kedalam kelas sentimen positif.

Pada data ulasan berbahasa Indonesia diketahui bahwa, terdapat 202 data positif yang benar terprediksi masuk kedalam kelas sentimen positif dan 4 data positif yang terprediksi masuk kedalam kelas sentimen negatif, serta 23 data negatif yang benar terprediksi masuk kedalam kelas sentimen negatif dan 32 data negatif yang terprediksi masuk kedalam kelas sentimen positif.

Tabel *confusion matrix* juga dapat digunakan untuk menghitung nilai ukuran evaluasi model yang lainnya, seperti *error rate*, *recall/sensitivity*, *specificity* dan *precision* yang pada Tabel 4.10.

Tabel 4.10 Nilai Ukuran Evaluasi Model

Ukuran	**Bahasa Inggris**	**Bahasa Indonesia**
Akurasi	89,3%	86,2%
Error rate	10,7%	13,8%
Recall/sensitivity	100%	98%
Specificity	12,5%	41,8%
Precision	89,1%	89,8%

4.1.8 Visualisasi

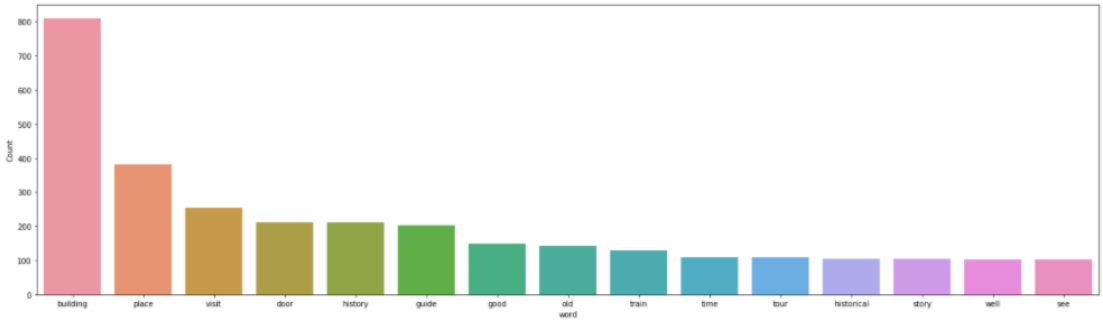
Tujuan visualisasi adalah untuk mengekstraksi informasi berupa topik yang paling sering dibicarakan/diulas oleh pengunjung Lawang Sewu, sehingga dari sekian banyak teks ulasan yang ada, dapat diambil informasi yang dianggap penting. Dalam penelitian ini visualisasi hasil dari analisis klasifikasi akan menggunakan *word cloud*. *Word cloud* adalah representasi dari suatu data yang menampilkan kumpulan kata-kata penting dan sering muncul dari data tersebut. Semakin besar kata itu muncul dalam *word cloud*, maka semakin besar pula frekuensi kemunculan kata itu dalam data.

4.1.8.1 Data Ulasan Berbahasa Inggris

4.1.8.1.1 Sentimen Positif

Data ulasan positif adalah hasil pelabelan yang masuk pada kelas positif menggunakan analisis sentimen berbasis Vader. Ulasan positif tersebut di identifikasi berdasarkan banyaknya frekuensi kata dalam ulasan. berikut adalah hasil visualisasi ulasan positif dari hasil ekstraksi informasi yang didapatkan dari ulasan-ulasan yang ditulis oleh pengunjung. *Script* untuk melihat kata yang sering muncul pada kelas sentimen positif untuk data ulasan berbahasa Inggris adalah sebagai berikut, berserta *output*nya pada Gambar 4.20.

```
#memplot kata-kata yang paling sering muncul
import string
table = str.maketrans('', '', string.punctuation)
strippedpos = [w.translate(table) for w in positive]
freq_words(strippedpos, 15)
```



Gambar 4.20 Frekuensi Kata pada Kelas Positif Data Berbahasa Inggris

Berdasar Gambar 4.20 diperoleh informasi bahwa pada kelas sentimen positif kata yang paling sering muncul adalah kata-kata *building, place, visit, door, history, guide, good, old, train, time, tour, historical, story, well dan see*. Kata-kata tersebut ditampilkan dalam visualisasi word cloud dengan *script* seperti berikut, dengan outut ditampilkan pada Gambar 4.21.

```
#wordcloud positif
from wordcloud import WordCloud
from wordcloud import ImageColorGenerator
all_text_positive = ' '.join(str(word) for word in strippedpos)
wordcloud = WordCloud(max_font_size=260, max_words=50, width=
1000, height=1000, mode='RGBA', background_color='white').genera
te(all_text_positive)
plt.figure(figsize=(15,8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.margins(x=0, y=0)
plt.show()
```

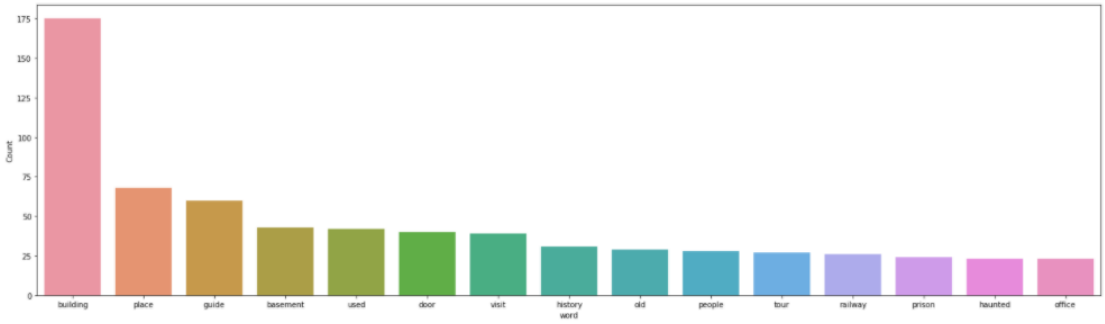
Gambar 4.21 *Word Cloud* Kelas Positif Data Berbahasa Inggris

4.1.8.1.2 Sentimen negatif

Data ulasan negatif adalah hasil pelabelan yang masuk pada kelas negatif menggunakan analisis sentimen berbasis Vader. Ulasan negatif tersebut diidentifikasi berdasarkan banyaknya frekuensi kata dalam ulasan. berikut adalah hasil visualisasi

ulasan pnegatif dari hasil ekstraksi informasi yang didapatkan dari ulasan-ulasan yang ditulis oleh pengunjung. *Script* untuk melihat kata yang sering muncul pada kelas sentimen negatif untuk data ulasan berbahasa Inggris adalah sebagai berikut, berserta *output*nya pada Gambar 4.22.

```
#memplot kata-kata yang paling sering muncul
import string
table = str.maketrans('', '', string.punctuation)
strippedneg = [w.translate(table) for w in negative]
freq_words(strippedneg, 15)
```



Gambar 4.22 Frekuensi Kata pada Kelas Negatif Data Berbahasa Inggris

Berdasar Gambar 4.22 diperoleh informasi bahwa pada kelas sentimen negatif kata yang paling sering muncul adalah kata-kata *building, place, guide, basement, used, door, visit, history, old, people, tour, railway, prison, haunted dan office*. Kata-kata tersebut ditampilkan dalam visualisasi word cloud dengan *script* seperti berikut, dengan hasil *word cloud* ditampilkan pada Gambar 4.23.

```
#wordcloud negatif
from wordcloud import WordCloud
from wordcloud import ImageColorGenerator
all_text_negative = ' '.join(str(word) for word in strippedneg)
wordcloud = WordCloud(max_font_size=260, max_words=50, width=
1000, height=1000, mode='RGBA', background_color='white').genera
te(all_text_negative)
plt.figure(figsize=(15,8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.margins(x=0, y=0)
plt.show()
```

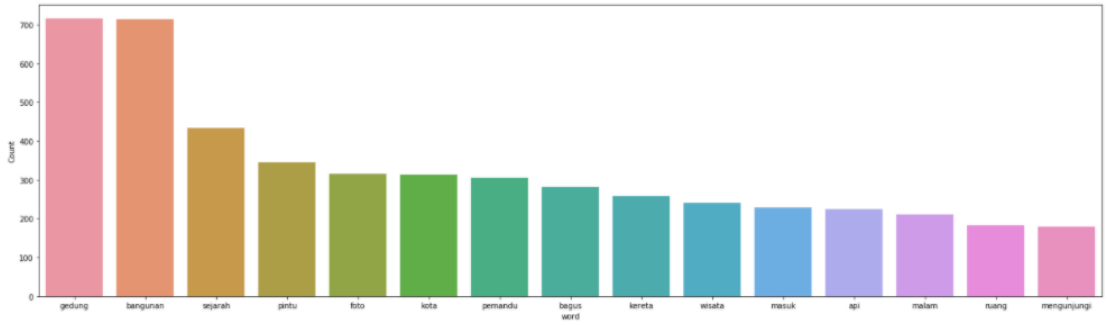
Gambar 4.23 *Word Cloud* Kelas Negatif Data Berbahasa Inggris

4.1.8.2 Data Ulasan Berbahasa Indonesia

4.1.8.2.1 Sentimen Positif

Data ulasan positif adalah hasil pelabelan yang masuk pada kelas positif menggunakan analisis sentimen. Ulasan positif tersebut di identifikasi berdasarkan banyaknya frekuensi kata dalam ulasan. berikut adalah hasil visualisasi ulasan positif dari hasil ekstraksi informasi yang didapatkan dari ulasan yang ditulis pengunjung. *Script* untuk melihat kata yang sering muncul pada kelas sentimen positif untuk data ulasan berbahasa Indonesia adalah sebagai berikut, berserta *output* diagram pada Gambar 4.24.

```
#memplot kata-kata yang paling sering muncul
import string
table = str.maketrans('', '', string.punctuation)
strippedpos = [w.translate(table) for w in positive]
freq_words(strippedpos, 15)
```



Gambar 4.24 Frekuensi Kata pada Kelas Positif Data Berbahasa Indonesia

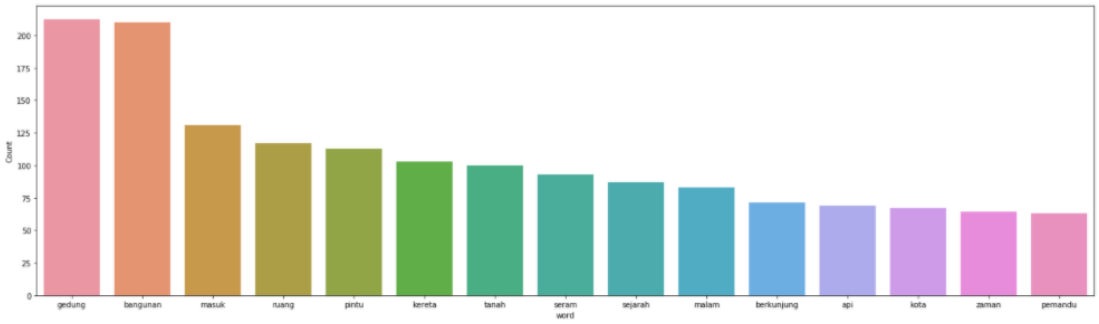
Berdasar Gambar 4.24 diperoleh informasi bahwa pada kelas sentimen positif kata yang paling sering muncul adalah kata-kata gedung, bangunan, sejarah, pintu, foto, kota, pemandu, bagus, kereta, wisata, masuk, api, malam, ruang dan mengunjungi. Kata-kata tersebut ditampilkan dalam visualisasi *word cloud* seperti berikut, dengan hasilnya ditampilkan pada Gambar 4.25.

Gambar 4.25 *Word Cloud* Kelas Positif Data Berbahasa Indonesia

4.1.8.2.2 Sentimen Negatif

Data ulasan negatif adalah hasil pelabelan yang masuk pada kelas negatif menggunakan analisis sentimen berbasis Vader. Ulasan negatif tersebut diidentifikasi berdasarkan banyaknya frekuensi kata dalam ulasan. berikut adalah hasil visualisasi ulasan pnegatif dari hasil ekstraksi informasi yang didapatkan dari ulasan-ulasan yang ditulis oleh pengunjung. *Script* untuk melihat kata yang sering muncul pada kelas sentimen negatif untuk data ulasan berbahasa Indonesia adalah sebagai berikut, berserta *output*nya pada Gambar 4.26.

```
#memplot kata-kata yang paling sering muncul
import string
table = str.maketrans('', '', string.punctuation)
strippedpos = [w.translate(table) for w in positive]
freq_words(strippedpos, 15)
```



Gambar 4.26 Frekuensi Kata pada Kelas Negatif Data Berbahasa Indonesia

Berdasar Gambar 4.26 diperoleh informasi bahwa pada kelas sentimen negatif kata yang paling sering muncul adalah kata-kata gedung, bangunan, masuk, ruang, pintu, kereta, tanah, seram, sejarah, malam, berkunjung, api, kota, zaman dan pemandu. Kata-kata tersebut ditampilkan dalam visualisasi *word cloud* seperti berikut, dengan hasil yang ditampilkan pada Gambar 4.27.

Gambar 4.27 *Word Cloud* Kelas Negatif Data Berbahasa Indonesia

4.2 Pembahasan

Berikut pembahasan hasil analisis yang telah dilakukan, setelah melakukan pengumpulan data menggunakan teknik *web scraping* didapatkan dua data ulasan pengunjung Lawang Sewu, yaitu data ulasan berbahasa Inggris dengan banyak data 652 ulasan dan data ulasan berbahasa Indonesia dengan banyak data 1303 ulasan.

Analisis dalam penelitian ini berbantuan *software* Anaconda Navigator menggunakan Jupyter Notebook dengan bahasa pemrograman Python, tahapan dalam analisis ini diantaranya adalah analisis deskriptif, *text preprocessing*, pembobotan kata, pelabelan data, klasifikasi data dan visualisasi.

Pada analisis deskriptif diperoleh informasi, bahwa dari kedua data, baik data ulasan berbahasa Inggris dan Indonesia, pengunjung Lawang Sewu lebih banyak memberikan ulasan dengan penilaian/rating yang baik, yakni untuk data ulasan berbahasa Inggris diperoleh rating 1 sampai 5 secara berturut-turut sebanyak 5, 19, 109, 305 dan 214 ulasan. Sedangkan untuk data ulasan berbahasa Indonesia diperoleh penilaian 1 sampai 5 secara beturut-turut sebanyak 3, 11, 213, 620 dan 456 ulasan.

Tahap *preprocessing* adalah proses pembersihan, dalam proses ini sendiri juga terdapat beberapa tahapan yakni diantaranya *case folding, cleansing, remove number*,

remove punctuation, remove short word, remove single char, tokenizing, spell normalization, filtering dan *remove extend stopword*.

Setelah didapatkan data yang terstruktur dan bersih, dilanjutkan dengan proses pelabelan kelas sentimen, yaitu melabelkan data kedalam dua kelas sentimen, positif dan negatif. Dari proses pelabelan ini didapatkan hasil perolehan banyaknya data ulasan yang masuk ke kelas sentimen positif dan negatif untuk data ulasan berbahasa Inggris sebanyak 555 sentimen positif dan 97 sentimen negatif, sedangkan untuk data berbahasa Indonesia didapat 1017 data masuk ke kelas sentimen positif dan 286 data masuk ke kelas sentimen negatif.

Sebelum melanjutkan analisis ke tahap klasifikasi, data ulasan perlu terlebih dahulu dipartisi menjadi data latih dan data uji, pembagian data ini adalah sebesar 80% untuk data latih dan 20% untuk data uji. Data latih digunakan untuk melatih algoritma klasifikasi dalam mempelajari ciri-ciri dan perbedaan dari kedua kelas sentimen. Data uji digunakan untuk melihat presentase keberhasilan dalam melakukan klasifikasi dengan benar.

Klasifikasi yang dilakukan dalam penelitian ini adalah membandingkan lima metode klasifikasi yang berbeda, diantaranya *Support Vector Mechine, Naïve Bayaes, Random Forest, Decision Tree* dan *Logistic Regression*. Dari komparasi nilai akurasi kelima metode klasifikasi, diketahui bahwa SVM adalah metode terbaik yang dignakan untuk pengklasifikasian data ulasan pengunjung objek wisata Lawang Sewu dilihat dari nilai akurasinya yang paling baik (besar). Nilai akurasi dengan metode *Support Vector Mechine* sebesar 89,3% untuk data ulasan berbahasa Inggris dan 86,2% untuk data ulasan berbahasa Indonesia.

Data hasil klasifikasi divisualisasikan dengan *word cloud*, dalam *word cloud* diketahui kata-kata mana saja yang sering muncul, dan dari *word cloud* didapatkan informasi seperti, penilaian positif dari pengunjung diantaranya tentang nilai sejarah yang terkandung, arsitektur gedung yang unik dan indah, bangunan yang bersih dan terawat, cerita menarik dari pemandu dan letak Lawang Sewu yang berada di tengah kota. Sedangkan beberapa penilaian negatif dari pengunjung diantaranya adalah suasana gedung yang gelap sehingga terkesan seram, angker dan mistis apalagi jika

dikunjungi pada malam hari, lantai ruang bawah tanah yang berair serta parkir yang selalu penuh karena ketersediaan lahan yang terbatas. Dari beberapa penilaian negatif dapat dilakukan tindak lanjut dengan perbaikan Lawang Sewu agar menghilangkan kesan seram tanpa mengurangi nilai sejarah, menambah penerangan dan menambah lahan parkir untuk pengunjung. Informasi tersebut dapat berguna bagi Dinas Kebudayaan dan Pariwisata Kota Semarang, pengelola objek wisata Lawang Sewu maupun pihak lain yang membutuhkan.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil analisis dan pembahasan yang telah dijelaskan pada bagian sebelumnya, maka diperoleh beberapa simpulan sebagai berikut.

1. Berdasar analisis deskriptif terhadap jumlah penilaian/rating yang diberikan oleh pengunjung Lawang Sewu pada situs TripAdvisor, kedua data ulasan berbahasa Inggris dan berbahasa Indonesia, menunjukan hasil bahwa pengunjung objek wisata Lawang Sewu mayoritas memberikan penilaian yang baik. Dari 652 data ulasan berbahasa Inggris diperoleh 214 pengunjung memberikan *rating excellent*, 305 pengunjung memberikan *rating very good*, 109 pengunjung memberikan rating average, 19 pengunjung memberikan *rating poor* dan 5 pengunjung memberikan *rating terrible*. Sedangkan untuk data ulasan berbahasa Indonesia, dari 1303 data ulasan, terdapat 620 pengunjung memberikan penilaian luar biasa, 456 pengunjung memberikan penilaian sangat bagus, 213 pengunjung memberikan penilaian rata-rata, 11 pengunjung memberikan penilaian buruk dan 3 pengunjung memberikan penilaian sangat buruk.
2. Penerapan metode klasifikasi *Support Vector Mechine*, *Naïve Bayaes*, *Random Forest*, *Decision Tree* dan *Logistic Regression* dalam pengklasifikasian sentimen positif dan sentimen negatif pada data ulasan pengunjung objek wisata Lawang Sewu, menghasilkan nilai akurasi untuk masing-masing metode berturut-turut adalah 89,3%, 87,7%, 87,7%, 76,3% dan 87,7% untuk data ulasan berbahasa Inggris, 86,2%, 78,9%, 80,0%, 72,4% dan 80,5% untuk data ulasan berbahasa Indonesia. Nilai akurasi tersebut menunjukkan bahwa metode *Support Vector Mechine* paling baik untuk mengklasifikasikan data ulasan pengunjung Lawang Sewu karena menghasilkan nilai akurasi tertinggi kedeua data ulasan, yaitu 89,3% untuk data ulasan berbahasa Inggris dan 86,2% untuk data ulasan berbahasa Indonesia.

3. Pengetahuan yang didapatkan dari analisis sentimen dan klasifikasi pada data ulasan pengunjung objek wisata Lawang Sewu tersebut adalah, dari kedua data diketahui bahwa ulasan/*review* pengunjung lebih banyak yang bersentimen positif.
4. Visualisasi data dari hasil klasifikasi yang dilakukan.
   a. *Word cloud* sentimen positif untuk data ulasan berbahasa Inggris, diketahui ada beberapa kata-kata yang paling sering muncul diantaranya: *building*, *place*, *visit*, *door*, *history*, *guide*, *good*, *old*, *train*, *tour*, *historical*, *story* dan *well*.

   b. *Word cloud* sentimen negatif untuk data ulasan berbahsa Inggris, diketahui ada beberapa kata-kata yang paling sering muncul diantaranya: *building*, *place*, *guide*, *basement*, *used*, *door*, *good*, *visit*, *history*, *old*, *tour*, *railway*, *prison* dan *office*.

c. *Word cloud* sentimen positif untuk data ulasan berbahasa Indonesia, diketahui ada beberapa kata-kata yang paling sering muncul diantaranya: gedung, bangunan, sejarah, pintu, foto, kota, pemandu, kereta, wisata, menarik dan mengunjungi.

d. *Word cloud* sentimen negatif untuk data ulasan berbahasa Indonesia, diketahui ada beberapa kata-kata yang paling sering muncul diantaranya: gedung, bangunan, masuk, ruang, pintu, kereta, tanah, seram, malam dan parkir.

Berdasar visualisasi data *word cloud*, beberapa penilaian positif dari pengunjung diantaranya tentang nilai sejarah, arsitektur gedung, bangunan yang bersih dan terawat, cerita menarik dari pemandu dan letak Lawang Sewu di tengah kota. Sedangkan penilaian negatif dari pengunjung diantaranya adalah suasana gedung seram apalagi jika dikunjungi malam hari, lantai ruang bawah tanah yang berair dan parkir yang sempit.

5.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan maka ada beberapa saran yang disampaikan diantaranya sebagai berikut.

1. Dalam penelitian ini menggunakan data dengan tiga variabel, yaitu penilaian/*rating*, nama/*name* dan ulasan/*review*. Untuk penelitian selanjuutnya, dapat menggunakan variabel yang lebih banyak, agar dapat semakin banyak variabel yang dapat dianalisis deskriptif.
2. Dalam penalitian ini menggunakan dua data, yaitu data ulasan berbahasa Inggris dan data berbahasa Indonesia. Untuk penelitian selanjutnya, dapat menambah menggunakan data dengan bahasa asing lainnya.
3. Dalam penalitian ini menggunakan data ulasan yang diambil sejak januari 2011 sampai maret 2020. Untuk penelitian selanjutnya, dapat memilih objek penelitian yang memiliki ketersedian data yang lebih banyak dan memilih batasan periode yang lebih kecil.
4. Dari hasil penelitian ini didapatkan informasi mengenai penilaian negatif yang diberikan oleh pengunjung objek wisata Lawang Sewu, tindak lanjut yang dapat dilakukan oleh Dinas Kebudayaan dan Pariwisata Kota Semarang, pengelola objek wisata Lawang Sewu maupun pihak lain yang membutuhkan, untuk mengatasi penilaian negatif tersebut antara lain dengan melakukan perbaikan Lawang Sewu agar menghilangkan kesan seram tanpa mengurangi nilai sejarah, menambah penerangan dan menambah lahan parkir untuk pengunjung.

DAFTAR PUSTAKA

Afifah, Adilla. 2015. "Respon Pelanggan Pada Situs Tripadvisor.Com Sebagai Bentuk Cyber Public Relations The Premiere Hotel Pekanbaru." *Jom FISIP Universitas Riau* 2(2):2.

Agung, I. Gusti Ngurah. 2000. "Analisis Statistik Sederhana Untuk Pengambilan Keputusan." *Jurnal Populasi Kependudukan Dan Kebijakan Universitas Gadjah Mada* 11(2):77.

Alif, Faris Zharfan. 2020. "Ekstraksi Fitur Untuk Pemilihan Topik Spesifik Review Film Dalam Menghasilkan Aspect-Based Sentiment Analysis." *Univsersitas Sumatera Utara* 6.

ASH. 2020. "TFIDF Separate for Each Label." *Stack Overflow*. Retrieved June 2, 2020 (https://stackoverflow.com/questions/60686556/tfidf-separate-for-each-label).

Bedi, Gunjit. 2018. "A Guide to Text Classification(NLP) Using SVM and Naive Bayes with Python." *Medium*. Retrieved July 27, 2002 (https://medium.com/@bedigunjit/simple-guide-to-text-classification-nlp-using-svm-and-naive-bayes-with-python-421db3a72d34).

Devy, Helln Angga, and R. B. Soemanto. 2017. "Pengembangan Obyek Dan Daya Tarik Wisata Alam Sebagai Daerah Tujuan Wisata Di Kabupaten Karanganyar (Studi Kasus Obyek Wisata Air Terjun Jumog Di Kawasan Wisata Desa Berjo, Kecamatan Ngargoyoso, Kabupaten Karanganyar)." *Jurnal DILEMA Sosiologi Universitas Sebelas Maret* 32(1):35.

DISBUDPAR. 2019. "Wisata Warisan Budaya Lawang Sewu." *Dinas Kebudayaan Dan Pariwisata Kota Semarang*. Retrieved April 2, 2020 (http://pariwisata.semarangkota.go.id/lawang-sewu/).

Evanmartua. 2020. "Twitter COVID19 Indonesia Sentiment Analysis Lexicon Based." *GitHub*. Retrieved (https://github.com/evanmartua34/Twitter-COVID19-Indonesia-Sentiment-Analysis---Lexicon-Based).

Giuseppegambino. 2020. "Scraping TripAdvisor with Python 2020." *GitHub*.

Retrieved March 20, 2020 (https://github.com/giuseppegambino/Scraping-TripAdvisor-with-Python-2020).

Handayani, Fitri, and Feddy Setio Pribadi. 2015. "Implementasi Algoritma Naive Bayes Classifier Dalam Pengklasifikasian Teks Otomatis Pengaduan Dan Pelaporan Masyarakat Melalui Layanan Call Center 110." *Jurnal Teknik Elektro Universitas Negeri Semarang* 7(1):20.

Imron, Ali. 2019. "Analisis Sentimen Terhadap Tempat Wisata Di Kabupaten Rembang Menggunakan Metodde Naïve Bayes Classifier." *Jurnal Teknik Informatika Univaersitas Islam Indonesia*.

Josi, Ahmat, Leon Andretti Abdillah, and Suryayusra. 2014. "Penerapan Teknik Web Scraping Pada Mesin Pencari Artikel Ilmiah." *Jurnal Ilmu Komputer Universitas Bina Darma*.

Khotimnr. 2019. "SentiStrengthID." *GitHub*. Retrieved April 20, 2002 (https://github.com/khotimnr/SentiStrengthID/blob/master/Sentiment_Analysis_(Solution).ipynb).

Mardi, Yuli. 2017. "Data Mining : Klasifikasi Menggunakan Algoritma C4.5." *Jurnal Edik Informatika STKIP PGRI Sumatera Barat* 2(2):216.

Menarianti, Ika. 2015. "Klasifikasi Data Mining Dalam Menentukan Pemberian Kredit Bagi Nasabah Koperasi." *Jurnal Ilmiah Teknosains* 1(1):40.

Meyers, Koen. 2009. *Ekowisata: Panduan Dasar Pelaksanaan*. Jakarta: UNESCO Digital Library.

Mishra, Madhav. 2020. "Python Implementation of SVM, Logistics Regression, Naive Bayes, Decision Tree, Random Forest Using Scikit-Learn (Just 3 Line of Code)." *Medium*. Retrieved July 22, 2002 (https://medium.com/analytics-vidhya/python-implementation-of-svm-logistics-regression-naive-bayes-decision-tree-random-forest-1f8a5755c37b).

Purbo, Onno W. 2019. *Text Mining Analisis MedSos, Kekuatan Brand & Intelejen Di Internet*. edited by A. A. Christian. Yogyakarta: Andi.

Rahman, M. Fadl., M. Ilha. Darmawidjadja, and Dion Alamsah. 2017. "Klasifikasi Untuk Diagnosa Diabetes Menggunakan Metode Bayesian Regularization

Neural Network (Rbnn)." *Jurnal Informatika Universitas Padjadjaran* 11(1):40.

Ramli, Desi Yuniarti, and Rito Goejantoro. 2013. "Perbandingan Metode Klasifikasi Regresi Logistik Dengan Jaringan Saraf Tiruan (Studi Kasus: Pemilihan Jurusan Bahasa Dan IPS Pada SMAN 2 Samarinda Tahun Ajaran 2011/2012)." *Jurnal EKSPONENSIAL Universitas Mulawarman* 4(1):17.

Rozi, Imam Fathrur, Sholeh Hadi Pramono, and Erfan Achmad Dahlan. 2012. "Implementasi Opinion Mining (Analisis Sentimen) Untuk Ekstraksi Data Opini Publik Pada Perguruan Tinggi." *Jurnal EECCIS Universitas Brawijaya* 6(1):37.

Setio, Panji Bimo Nugroho, Dewi Retno Sari Saputro, and Bowo Winarno. 2020. "Klasifikasi Dengan Pohon Keputusan Berbasis Algoritme C4.5." *Prosiding Seminar Nasional Matematika*.

Setya, Mayang Vini. 2017. "Strategi Dinas Kebudayaan Dan Pariwisata Kota Semarang Dalam Upaya Mengembangkan Pariwisata Kota Semarang." *Jurnal Ilmu Pemerintahan Undip* 6.

Sugiana, Owo. 2003. *Membuat Aplikasi Bisnis Menggunakan Bahasa Python Dan Database Berbasis SQL*. Jakarta.

Sugiyono. 2009. *Metode Penelitian Pendidikan Pendekatan Kuantitatif, Kualitatif, Dan R&D*. Bandung: Alfabeta.

Suryono, Sigit, Ema Utami, and Emba Taufiq Luthfi. 2018. "Klasifikasi Sentimen Pada Twitter Dengan Naive Bayes Classifier." *ANGKASA-Jurnal Ilmiah Bidang Teknologi* 10(1):91.

Suyanto. 2019. *Data Mining Untuk Klasifikasi Dan Klasterisasi Data*. Bandung: Informatika.

TripAdvisor. 2018. "TripAdvisor." Retrieved (https://play.google.com/store/apps/details?id=com.tripadvisor.tripadvisor&hl=in).

Vulandari, Retno Tri. 2017. *Data Mining Teori Dan Aplikasi Rapidminer*. Yogyakarta: Gava Media.

Yunus, Muhammad. 2020a. "Text Preprocessing Menggunakan Pandas, NLTK Dan Sastrawi Untuk Large Dataset." *Medium*. Retrieved June 12, 2020

(https://medium.com/@yunusmuhammad007/text-preprocessing-menggunakan-pandas-nltk-dan-sastrawi-untuk-large-dataset-5fb3c0a88571).

Yunus, Muhammad. 2020b. "TF-IDF (Term Frequency-Inverse Document Frequency) : Representasi Vector Data Text." *Medium*. Retrieved April 17, 2020 (https://medium.com/@yunusmuhammad007/tf-idf-term-frequency-inverse-document-frequency-representasi-vector-data-text-2a4eff56cda).

LAMPIRAN

Lampiran 1. *Script Scrap* Data Ulasan Berbahasa Inggris

Scraping ulasan Lawang Sewu pada situs TripAdvisor

1. Mengimport library

In [1]:
```
import csv
import time
from selenium import webdriver
from selenium.common.exceptions import NoSuchElementException
```

2. Mengimport webdriver dan memasukan alamat website yang akan discarp

In [2]:
```
# mengimport webdriver
from selenium.webdriver.chrome.options import Options
chrome_options = Options()
chrome_options.add_argument("--window-size=1920,1080")
chrome_options.add_experimental_option( "prefs",{'profile.managed_default_content_settings.images': 2})
driver = webdriver.Chrome('C:/chromedriver_win32/chromedriver.exe', chrome_options=chrome_options)

# masukan alamat website TripAdvisor Lawang Sewu
driver.get("https://www.tripadvisor.com/Attraction_Review-g297712-d379332-Reviews-Lawang_Sewu_Building-Semarang_Central_Java_Java
```

```
C:\Users\Acer\anaconda3\lib\site-packages\ipykernel_launcher.py:6: DeprecationWarning: use options instead of chrome_options
```

3. Scrap

untuk data yang akan diambil hanya data nama, penilaian dan ulasan

In [ ]:
```
# fungsi untuk memeriksa button pada halaman, untuk menghindari masalah miss-click
def check_exists_by_xpath(xpath):
    try:
        driver.find_element_by_xpath(xpath)
    except NoSuchElementException:
        return False
    return True

#time.sleep(2)
#driver.find_element_by_xpath('//span[contains(@class, "ExpandableReview")]').click()
time.sleep(2)

# memmbuka file untuk menyimpan ulasan
csvFile = open("scrap inggris.csv", 'a', encoding="utf-8")
csvWriter = csv.writer(csvFile)

# ubah nilai di dalam tanda kurung untuk menyimpan lebih banyak atau sedikit ulasan
for i in range(0,6000):
    if (check_exists_by_xpath("//span[@class='taLnk ulBlueLinks']")):

        # to expand the review
        driver.find_element_by_xpath("//span[@class='taLnk ulBlueLinks']").click()
        time.sleep(5)
    container = driver.find_elements_by_xpath("//div[@class='Dq9MAugU T870kzTX LnVzGwUB']")
    num_page_items = len(container);
    #print(container)
    print(num_page_items)
    for j in range(num_page_items):
        # to save the rating
        string = container[j].find_element_by_xpath(".//span[contains(@class, 'ui_bubble_rating bubble_')]").get_attribute("class
        # to save in a csv file readable the star and the review [Ex: 50,"I love this place"]
        data = string.split("_")
        rating = int(data[3])/10
        poster_name = container[j].find_element_by_xpath(".//a[contains(@class,'ui_header_link _1r_My98y')]").text.replace("\n","
        #print(poster_name)

        review_text = container[j].find_element_by_xpath(".//q[@class='IRsGHoPm']").text.replace("\n", "")

        print(str(rating) + "  " + poster_name + " " + review_text)
        csvWriter.writerow([str(rating),poster_name,review_text.encode('ascii', 'ignore').decode('ascii') ])

    # untuk mengubah halaman pada website
    if check_exists_by_xpath(".//a[contains(@class, 'ui_button nav next primary ')]"):
        driver.find_element_by_xpath(".//a[contains(@class, 'ui_button nav next primary ')]").click()
    #driver.find_element_by_xpath('//a[contains(@class, "next")]').click()
    time.sleep(6)

driver.close()
```

```
pite Lawang Sewu's reputation - I didn't spot…
4.0  deddy p Want to come hassle free? Use public transport (Taxi). Come on weekdays. U wanna get a good photos, come by nigh
t time. Need a help for a good photo spot, you may asked a kinda of tour guide inside. No parking space.....
```

Lampiran 2. *Script Scrap* Data Ulasan Berbahasa Indonesia

Scraping ulasan Lawang Sewu pada situs TripAdvisor

1. Mengimport library

In [1]:
```
import csv
import time
from selenium import webdriver
from selenium.common.exceptions import NoSuchElementException
```

2. Mengimport webdriver dan memasukan alamat website yang akan discarp

In [3]:
```
# mengimport webdriver
from selenium.webdriver.chrome.options import Options
chrome_options = Options()
chrome_options.add_argument("--window-size=1920,1080")
chrome_options.add_experimental_option( "prefs",{'profile.managed_default_content_settings.images': 2})
driver = webdriver.Chrome('C:/chromedriver_win32/chromedriver.exe', chrome_options=chrome_options)

# masukan alamat website TripAdvisor Lawang Sewu
driver.get("https://www.tripadvisor.co.id/Attraction_Review-g297712-d379332-Reviews-Lawang_Sewu_Building-Semarang_Central_Java_Ja
```

```
C:\Users\Acer\anaconda3\lib\site-packages\ipykernel_launcher.py:6: DeprecationWarning: use options instead of chrome_options
```

3. Scrap

untuk data yang akan diambil hanya data nama, penilaian dan ulasan

In [4]:
```
# fungsi untuk memeriksa button pada halaman, untuk menghindari masalah miss-click
def check_exists_by_xpath(xpath):
    try:
        driver.find_element_by_xpath(xpath)
    except NoSuchElementException:
        return False
    return True

#time.sleep(2)
#driver.find_element_by_xpath('//span[contains(@class, "ExpandableReview")]').click()
time.sleep(2)

# memmbuka file untuk menyimpan ulasan
csvFile = open("uji scrap.csv", 'a', encoding="utf-8")
csvWriter = csv.writer(csvFile)

# ubah nilai di dalam tanda kurung untuk menyimpan lebih banyak atau sedikit ulasan
for i in range(0,6000):
    if (check_exists_by_xpath("//span[@class='taLnk ulBlueLinks']")):

        # to expand the review
        driver.find_element_by_xpath("//span[@class='taLnk ulBlueLinks']").click()
        time.sleep(5)

    container = driver.find_elements_by_xpath("//div[@class='Dq9MAugU T870kzTX LnVzGwUB']")
    num_page_items = len(container);
    #print(container)
    print(num_page_items)
    for j in range(num_page_items):
        # to save the rating
        string = container[j].find_element_by_xpath(".//span[contains(@class, 'ui_bubble_rating bubble_')]").get_attribute("class
        # to save in a csv file readable the star and the review [Ex: 50,"I love this place"]
        data = string.split("_")
        rating = int(data[3])/10
        poster_name = container[j].find_element_by_xpath(".//a[contains(@class,'ui_header_link _1r_My98y')]").text.replace("\n","
        #print(poster_name)

        review_text = container[j].find_element_by_xpath(".//q[@class='IRsGHoPm']").text.replace("\n", "")
        print(str(rating) + "  " + poster_name + " " + review_text)
        csvWriter.writerow([str(rating),poster_name,review_text.encode('ascii', 'ignore').decode('ascii') ])

    # untuk mengubah halaman pada website
    if check_exists_by_xpath(".//a[contains(@class, 'ui_button nav next primary ')]"):
        driver.find_element_by_xpath(".//a[contains(@class, 'ui_button nav next primary ')]").click()
    #driver.find_element_by_xpath('//a[contains(@class, "next")]').click()
    time.sleep(6)

driver.close()
```

```
5
5.0  Ignatius M Rame dengan pengunjung, letaknya dekat kemana-mana. Disekitaran Lawang Sewu juga banyak pilihan hotel dan dek
at Gereja Katedral
5.0  Deddy Cocok untuk wisata edukasi sejarah.kalau mengunjungi tempat sejarah bersama anak2 sebaiknya di siang hari
```

Lampiran 3. *Script* Analisis Data Berbahasa Inggris

Load data

Dataset yang digunakan adalah data ulasan pengunjung objek wisata Lawang Sewu Kota Semarang berbahasa Indonesia dari situs TripAdvisor yang tercatat sejak Januari 2011 hingga Maret 2020. Pengumpulan data tersebut dilakukan dengan cara web scraping

In [1]:
```
# load dataset kedalam dataframe

import pandas as pd
import numpy as np

df = pd.read_csv('C:/Users/Acer/Tugas Akhir/Scrap Lawang Sewu Inggris.csv')
print(df.shape)
```

```
(652, 3)
```

In [2]:
```
# melihat data

df.head()
```

Out[2]:

	rating	name	review
0	4.0	9999stephenh	This is an amazing building (collection of bui...
1	4.0	deddy p	Want to come hassle free? Use public transport...
2	5.0	TravelwithArben	It is quite interesting to experience such a h...
3	3.0	Stefanus Wijaya	This place previously known as haunted house, ...
4	4.0	Ubud98	Good, one of Semarang city's landmark. Suggest...

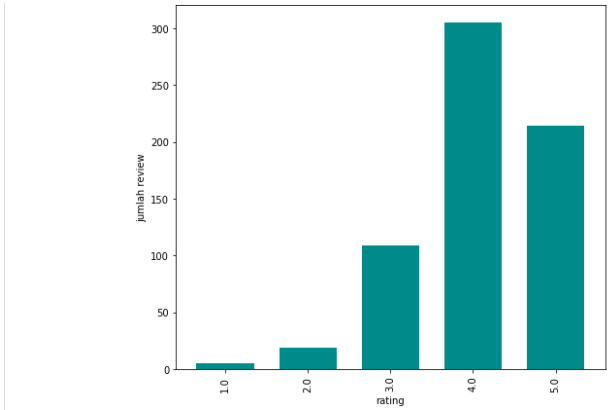
ANALISIS DESKRIPTIF

Melihat informasi secara umum tentang rating data ulasan pengunjung objek wisata Lawaang Sewu Kota Semarang

In [3]:
```
# melihat banyak ulasan dari tiap rating

df['rating'].value_counts()
```

Out[3]:
```
4.0    305
5.0    214
3.0    109
2.0     19
1.0      5
Name: rating, dtype: int64
```

In [4]:
```
# melihat distribusi banyak review berdasar rating

import matplotlib.pyplot as plt

_, ax1 = plt.subplots(figsize=(7,6))
stars_histogram = df['rating'].value_counts().sort_index()
stars_histogram.plot(kind='bar', width=0.7, color='darkcyan')
plt.xlabel('rating')
plt.ylabel('jumlah review')

plt.tight_layout()
plt.show()
```



PREPROCESSING

Pada tahap ini tujuannya adalah mengubah dataset supaya dapat diolah pada algoritma Machine Learning.

In [5]:
```
# menghapus baris jika ada data yang kosong

df = df.dropna()
print(df.shape)
```

```
(652, 3)
```

Case folding

In [6]:
```
# mengubah semua huruf dalam dokumen menjadi huruf kecil (lower text)

df['review'] = df['review'].str.lower()
df.head()
```

Out[6]:

	rating	name	review
0	4.0	9999stephenh	this is an amazing building (collection of bui...
1	4.0	deddy p	want to come hassle free? use public transport...
2	5.0	TravelwithArben	it is quite interesting to experience such a h...
3	3.0	Stefanus Wijaya	this place previously known as haunted house, ...
4	4.0	Ubud98	good, one of semarang city's landmark. suggest...

In [7]:
```
df.to_csv('result case folding.csv')
```

Filtering

In [8]:
```
# menghapus stopword

from nltk.corpus import stopwords
stop_words = stopwords.words('english')

# fungsi untuk menghapus stopword
def remove_stopwords(rev):
    rev_new = " ".join([i for i in rev if i not in stop_words])
    return rev_new
df.head()

# menghapus stopword dari teks
df['review'] = [remove_stopwords(r.split()) for r in df['review']]
df.head()
```

Out[8]:

	rating	name	review
0	4.0	9999stephenh	amazing building (collection buildings, actual...
1	4.0	deddy p	want come hassle free? use public transport (t...
2	5.0	TravelwithArben	quite interesting experience historic place ci...
3	3.0	Stefanus Wijaya	place previously known haunted house, already ...
4	4.0	Ubud98	good, one semarang city's landmark. suggest vi...

In [9]:
```
df.to_csv('result filtering.csv')
```

Cleansing

In [10]:
```
# pembersihan teks dari tab, new line, back slice, mention, link, hastag, URL

import string
import re #regex library

def remove_ulasan_special(text):
    # menghapus tab, new line, dan back slice
    text = text.replace('\\t'," ").replace('\\n'," ").replace('\\u'," ").replace('\\',"")
    # menghapus non ASCII (emoticon, chinese word, .etc)
    text = text.encode('ascii', 'replace').decode('ascii')
    # menghapus mention, link, hashtag
    text = ' '.join(re.sub("([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)"," ", text).split())
    # menghapus incomplete URL
    return text.replace("http://", " ").replace("https://", " ")

df['review'] = df['review'].apply(remove_ulasan_special)
df.head()
```

Out[10]:

	rating	name	review
0	4.0	9999stephenh	amazing building (collection buildings, actual...
1	4.0	deddy p	want come hassle free? use public transport (t...
2	5.0	TravelwithArben	quite interesting experience historic place ci...
3	3.0	Stefanus Wijaya	place previously known haunted house, already ...
4	4.0	Ubud98	good, one semarang city's landmark. suggest vi...

In [11]:
```
df.to_csv('result cleansing.csv')
```

Remove number

In [12]:
```
# menghapus angka

def remove_number(text):
    return  re.sub(r"\d+", "", text)

df['review'] = df['review'].apply(remove_number)
df.head()
```

Out[12]:

	rating	name	review
0	4.0	9999stephenh	amazing building (collection buildings, actual...
1	4.0	deddy p	want come hassle free? use public transport (t...
2	5.0	TravelwithArben	quite interesting experience historic place ci...
3	3.0	Stefanus Wijaya	place previously known haunted house, already ...
4	4.0	Ubud98	good, one semarang city's landmark. suggest vi...

In [13]:
```
df.to_csv('result remove number.csv')
```

Remove punctuation

In [14]:
```
# menghapus tanda baca

def remove_punctuation(text):
    return text.translate(str.maketrans("","",string.punctuation))

df['review'] = df['review'].apply(remove_punctuation)
df.head()
```

Out[14]:

	rating	name	review
0	4.0	9999stephenh	amazing building collection buildings actually...
1	4.0	deddy p	want come hassle free use public transport tax...
2	5.0	TravelwithArben	quite interesting experience historic place ci...
3	3.0	Stefanus Wijaya	place previously known haunted house already r...
4	4.0	Ubud98	good one semarang citys landmark suggest visit...

In [15]:
```
df.to_csv('result remove punctuation.csv')
```

Remove short words

In [16]:
```
# menghapus kata - kata pendek (kurang dari 3 huruf)

df['review'] = df['review'].apply(lambda x: ' '.join([w for w in x.split() if len(w)>2]))
df.head()
```

Out[16]:

	rating	name	review
0	4.0	9999stephenh	amazing building collection buildings actually...
1	4.0	deddy p	want come hassle free use public transport tax...
2	5.0	TravelwithArben	quite interesting experience historic place ci...
3	3.0	Stefanus Wijaya	place previously known haunted house already r...
4	4.0	Ubud98	good one semarang citys landmark suggest visit...

In [17]:
```
df.to_csv('result remove short words.csv')
```

Tokenizing

In [18]:
```
# pemisahan teks menjadi potongan-potongan kata yang disebut sebagai token

from nltk.tokenize import word_tokenize

# NLTK word tokenize
def word_tokenize_wrapper(text):
    return word_tokenize(text)

df['review'] = df['review'].apply(word_tokenize_wrapper)
df.head()
```

Out[18]:

	rating	name	review
0	4.0	9999stephenh	[amazing, building, collection, buildings, act...
1	4.0	deddy p	[want, come, hassle, free, use, public, transp...
2	5.0	TravelwithArben	[quite, interesting, experience, historic, pla...
3	3.0	Stefanus Wijaya	[place, previously, known, haunted, house, alr...
4	4.0	Ubud98	[good, one, semarang, citys, landmark, suggest...

In [19]:
```
df.to_csv('result tokenizing.csv')
```

Remove extend stopword

In [20]:
```
# memfilter untuk mengambil kata-kata penting dengan menggunakan algoritma stoplist (membuang kata kurang penting)
# stopword adalah kata umum yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna
# contoh stopword dalam bahasa Indonesia adalah “thing”, “many”, “also”, “evie ”

# import stopwords from NLTK
from nltk.corpus import stopwords

# dapatkan stopword dari NLTK stopword
stop_words = stopwords.words('english')

# menambahkan stopword tambahan
stop_words.extend(['thats', 'itll', 'youll'])

# menambahkan stopword dari file csv
txt_stopword = pd.read_csv('C:/Users/ACER/Tugas Akhir/stopword inggris.txt', names= ["stopwords"], header = None)

# mengkonversi kata stopword tambahan & stopword dari file csv ke list
stop_words.extend(txt_stopword["stopwords"][0].split(' '))

# mengkonversi list ke dictionary
stop_words = set(stop_words)

# menghapus stopword pada data
def stopwords_removal(words):
    return [word for word in words if word not in stop_words]

df['review'] = df['review'].apply(stopwords_removal)
df.head()
```

Out[20]:

	rating	name	review
0	4.0	9999stephenh	[amazing, building, collection, buildings, wel...
1	4.0	deddy p	[want, come, hassle, free, use, public, transp...
2	5.0	TravelwithArben	[interesting, experience, historic, place, pla...
3	3.0	Stefanus Wijaya	[place, previously, known, haunted, house, alr...
4	4.0	Ubud98	[good, landmark, suggest, visit, night, close]

In [21]:
```
df.to_csv('result remove extend stopword.csv')
```

In [22]:
```
df.to_csv('result preprocessing inggris.csv')
```

In [23]:
```
# load data hasil preprocessing kedalam dataframe

import pandas as pd
import numpy as np

df = pd.read_csv('C:/Users/Acer/Tugas Akhir/result preprocessing inggris.csv', usecols=["rating", "name", "review"])
df.columns = ["rating", "name", "review"]

df.head()
```

Out[23]:

	rating	name	review
0	4.0	9999stephenh	['amazing', 'building', 'collection', 'buildin...
1	4.0	deddy p	['want', 'come', 'hassle', 'free', 'use', 'pub...
2	5.0	TravelwithArben	['interesting', 'experience', 'historic', 'pla...
3	3.0	Stefanus Wijaya	['place', 'previously', 'known', 'haunted', 'h...
4	4.0	Ubud98	['good', 'landmark', 'suggest', 'visit', 'nigh...

In [24]:
```
# menggabungkan daftar token menjadi dokumen string tunggal

import ast

def join_text_list(texts):
    texts = ast.literal_eval(texts)
    return ' '.join([text for text in texts])

df["review"] = df["review"].apply(join_text_list)
df.head()
```

Out[24]:

	rating	name	review
0	4.0	9999stephenh	amazing building collection buildings well res...
1	4.0	deddy p	want come hassle free use public transport tax...
2	5.0	TravelwithArben	interesting experience historic place place ki...
3	3.0	Stefanus Wijaya	place previously known haunted house already r...
4	4.0	Ubud98	good landmark suggest visit night close

In [25]:
```
df.to_csv('result inggris.csv')
```

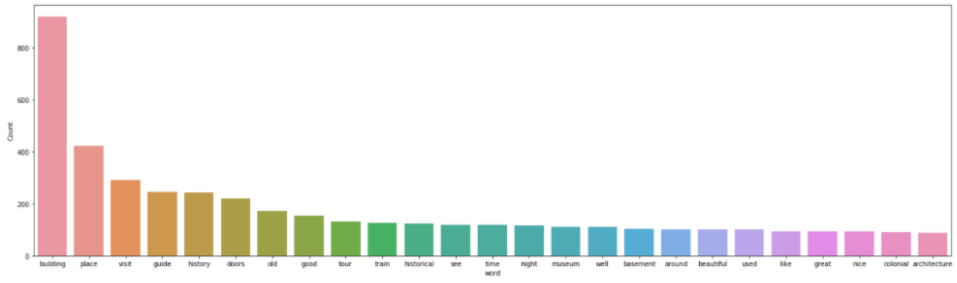
In [26]:
```
import difflib

#body_list = df['body'].tolist()
review_text_list = df['review'].tolist()

#body = body_list
reviews = review_text_list
s = difflib.SequenceMatcher(None, reviews).ratio()
print ("ratio:", s, "\n")
```

```
ratio: 0.0
```

Frekuensi & TFIDF

In [27]:
```
# fungsi untuk memplot term yang paling sering muncul

from nltk import FreqDist
import seaborn as sns

def freq_words(x, terms = 30):
  all_words = ' '.join([text for text in x])
  all_words = all_words.split()

  fdist = FreqDist(all_words)
  words_df = pd.DataFrame({'word':list(fdist.keys()), 'count':list(fdist.values())})

  # memilih 25 term yang paling sering muncul
  d = words_df.nlargest(columns="count", n = terms)
  plt.figure(figsize=(25,7))
  ax = sns.barplot(data=d, x= "word", y = "count")
  ax.set(ylabel = 'Count')
  plt.show()
```

In [28]:
```
# plot term yang paling sering muncul

freq_words(df['review'], 25)
```



In [29]:
```
# tf-idf

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
cvec = CountVectorizer(stop_words='english', min_df=1, max_df=.5, ngram_range=(1,2))
cvec
```

Out[29]:
```
CountVectorizer(max_df=0.5, ngram_range=(1, 2), stop_words='english')
```

In [30]:
```
# menghitung semua n-gram yang ditemukan di semua dokumen

from itertools import islice
from nltk.corpus import stopwords
global str

cvec.fit(review_text_list)
list(islice(cvec.vocabulary_.items(), 20))

len(cvec.vocabulary_)
```

Out[30]:
```
14765
```

In [31]:
```
cvec = CountVectorizer(stop_words='english', min_df=.0025, max_df=.5, ngram_range=(1,2))
cvec.fit(review_text_list)
len(cvec.vocabulary_)
```

Out[31]:
```
2563
```

In [32]:
```
cvec_counts = cvec.transform(review_text_list)
print('sparse matrix shape:', cvec_counts.shape)
print('nonzero count:', cvec_counts.nnz)
print('sparsity: %.2f%%' % (100.0 * cvec_counts.nnz / (cvec_counts.shape[0] * cvec_counts.shape[1])))
```

```
sparse matrix shape: (652, 2563)
nonzero count: 16175
sparsity: 0.97%
```

In [33]:
```
# menghitung frekuensi kemunculan term

occ = np.asarray(cvec_counts.sum(axis=0)).ravel().tolist()
counts_df = pd.DataFrame({'term': cvec.get_feature_names(), 'occurrences': occ})
counts_df.sort_values(by='occurrences', ascending=False).head(20)
```

Out[33]:

	term	occurrences
1727	place	423
2416	visit	291
984	guide	245
1109	history	242
641	doors	220
1588	old	171

In [34]:
```
# sekarang kita memiliki jumlah term untuk setiap dokumen kita dapat menggunakan Tfidf Transformer untuk menghitung
# bobot untuk setiap istilah dalam setiap dokumen

transformer = TfidfTransformer()
transformed_weights = transformer.fit_transform(cvec_counts)
transformed_weights
```

Out[34]:
```
<652x2563 sparse matrix of type '<class 'numpy.float64'>'
        with 16175 stored elements in Compressed Sparse Row format>
```

In [35]:
```
# melihat 20 term teratas dengan weight rata-rata tf-idf

weights = np.asarray(transformed_weights.mean(axis=0)).ravel().tolist()
weights_df = pd.DataFrame({'term': cvec.get_feature_names(), 'weight': weights})
weights_df.sort_values(by='weight', ascending=False).head(20)
```

Out[35]:

	term	weight
1727	place	0.050128
2416	visit	0.038192
1109	history	0.032755
641	doors	0.031293
984	guide	0.030098
927	good	0.026942

PELABELAN

In [36]:
```
# pelabelan data, data akan berlabel negatif jika nilai compound <0,0 dan akan berlabel positif jika nilai compound >=0,0

import nltk
from nltk.sentiment.vader import SentimentIntensityAnalyzer

sid = SentimentIntensityAnalyzer()
df['sentiment'] = df['review'].apply(lambda x: sid.polarity_scores(x))
def convert(x):
    if x < 0.0:
        return "negative"
    #elif x > 0.0:
        #return "positive"
    else:
        return "positive"
df['result'] = df['sentiment'].apply(lambda x:convert(x['compound']))

# df.groupby(['brand','result']).size()
# df.groupby(['brand','result']).count()
x = df.groupby(['review','name'])['result'].value_counts(normalize=True)

x = df.groupby(['review'])['result'].value_counts(normalize=True)
y = x.loc[(x.index.get_level_values(1) == 'negative')]
#print(y[y>0.2])
print(df.shape)
df.head()
```

```
(652, 5)
```

Out[36]:

	rating	name	review	sentiment	result
0	4.0	9999stephenh	amazing building collection buildings well res...	{'neg': 0.065, 'neu': 0.651, 'pos': 0.284, 'co...	positive
1	4.0	deddy p	want come hassle free use public transport tax...	{'neg': 0.0, 'neu': 0.656, 'pos': 0.344, 'comp...	positive
2	5.0	TravelwithArben	interesting experience historic place place ki...	{'neg': 0.145, 'neu': 0.449, 'pos': 0.406, 'co...	positive
3	3.0	Stefanus Wijaya	place previously known haunted house already r...	{'neg': 0.155, 'neu': 0.599, 'pos': 0.246, 'co...	positive
4	4.0	Ubud98	good landmark suggest visit night close	{'neg': 0.0, 'neu': 0.488, 'pos': 0.512, 'comp...	positive

In [37]:
```
# banyak data yang berlabel positif & negatif

df['result'].value_counts()
```

Out[37]:
```
positive    555
negative     97
Name: result, dtype: int64
```

METODE KLASIFIKASI

In [38]:
```
classificationLawangSewu = df
classificationLawangSewu = df.drop(columns=['rating', 'name', 'sentiment'])
classificationLawangSewu.head()
```

Out[38]:

	review	result
0	amazing building collection buildings well res...	positive
1	want come hassle free use public transport tax...	positive
2	interesting experience historic place place ki...	positive
3	place previously known haunted house already r...	positive
4	good landmark suggest visit night close	positive

In [39]:
```
# mengubah result

label = []
for index, row in classificationLawangSewu.iterrows():
    if row["result"] == 'positive':
        label.append(1)
    else:
        label.append(0)

classificationLawangSewu["label"] = label
classificationLawangSewu = classificationLawangSewu.drop(columns=['result'])
classificationLawangSewu.head()
```

Out[39]:

	review	label
0	amazing building collection buildings well res...	1
1	want come hassle free use public transport tax...	1
2	interesting experience historic place place ki...	1
3	place previously known haunted house already r...	1
4	good landmark suggest visit night close	1

In [40]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import model_selection, svm
from sklearn.metrics import accuracy_score
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder
```

In [41]:
```
# data training & data testing

Train_X, Test_X, Train_Y, Test_Y = model_selection.train_test_split(classificationLawangSewu['review'],classificationLawangSewu['

Encoder = LabelEncoder()
Train_Y = Encoder.fit_transform(Train_Y)
Test_Y = Encoder.fit_transform(Test_Y)

Tfidf_vect = TfidfVectorizer(max_features=5000)
Tfidf_vect.fit(classificationLawangSewu['review'])
Train_X_Tfidf = Tfidf_vect.transform(Train_X)
Test_X_Tfidf = Tfidf_vect.transform(Test_X)
```

In [42]:
```
len(Train_X)
```

Out[42]: 521

In [43]:
```
len(Test_X)
```

Out[43]: 131

Support Vector Mechine

In [44]:
```
# Support Vector Mechine

SVM = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='scale')
SVM.fit(Train_X_Tfidf,Train_Y)

predictions_SVM = SVM.predict(Test_X_Tfidf)

print("SVM Accuracy Score -> ",accuracy_score(predictions_SVM, Test_Y)*100)
```

```
SVM Accuracy Score ->  89.31297709923665
```

Naive Bayes

In [45]:
```
# Naive Bayes

from sklearn import naive_bayes
NB = naive_bayes.MultinomialNB()
NB.fit(Train_X_Tfidf,Train_Y)

predictions_NB = NB.predict(Test_X_Tfidf)

print("NB Accuracy Score -> ",accuracy_score(predictions_NB, Test_Y)*100)
```

```
NB Accuracy Score ->  87.78625954198473
```

Random Forest

In [95]:
```
# Random Forest

from sklearn.ensemble import RandomForestClassifier
RF = RandomForestClassifier()
RF.fit(Train_X_Tfidf,Train_Y)

predictions_RF = RF.predict(Test_X_Tfidf)

print("RF Accuracy Score -> ",accuracy_score(predictions_RF, Test_Y)*100)
```

```
RF Accuracy Score ->  87.78625954198473
```

Decision Tree

In [97]:
```
# Decision Tree

from sklearn.tree import DecisionTreeClassifier
DT = DecisionTreeClassifier()
DT.fit(Train_X_Tfidf,Train_Y)

predictions_DT = DT.predict(Test_X_Tfidf)

print("DT Accuracy Score -> ",accuracy_score(predictions_DT, Test_Y)*100)
```

```
DT Accuracy Score ->  76.33587786259542
```

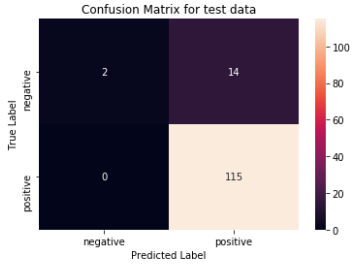
Logistic Regression

In [48]:
```
# Logistic Regression

from sklearn.linear_model import LogisticRegression
LR = LogisticRegression()
LR.fit(Train_X_Tfidf,Train_Y)

predictions_LR = LR.predict(Test_X_Tfidf)

print("LR Accuracy Score -> ",accuracy_score(predictions_LR, Test_Y)*100)
```

```
LR Accuracy Score ->  87.78625954198473
```

Confusion Matrix

In [49]:
```
# confusion matrix

from sklearn.metrics import confusion_matrix
import seaborn as sns

conf_mat = confusion_matrix(Test_Y, predictions_SVM)
class_label = ["negative", "positive"]
test = pd.DataFrame(conf_mat, index = class_label, columns = class_label)
sns.heatmap(test, annot = True,fmt="d")
plt.title("Confusion Matrix for test data")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.show()
```



VISUALISASI

Sentimen Negatif

In [51]:
```
LawangSewuNeg = classificationLawangSewu.loc[classificationLawangSewu['label']== 0]
LawangSewuNeg.head()
```

Out[51]:

	review	label
5	doors landmark built headquarters railway comp...	0
26	located center near old building railway offic...	0
27	name doors made friends curious prove started ...	0
36	famous horror site daytime see old colonial of...	0
43	considered mustsee nothing wander old building...	0

In [52]:
```
import difflib

#body_list = df['body'].tolist()
neg_text_list = LawangSewuNeg['review'].tolist()

#body = body_list
reviews_neg = neg_text_list
```

In [53]:
```
# stopword

from nltk.corpus import stopwords
english_stopwords = stopwords.words('english')
print(len(english_stopwords))
text_neg = str(neg_text_list)
```

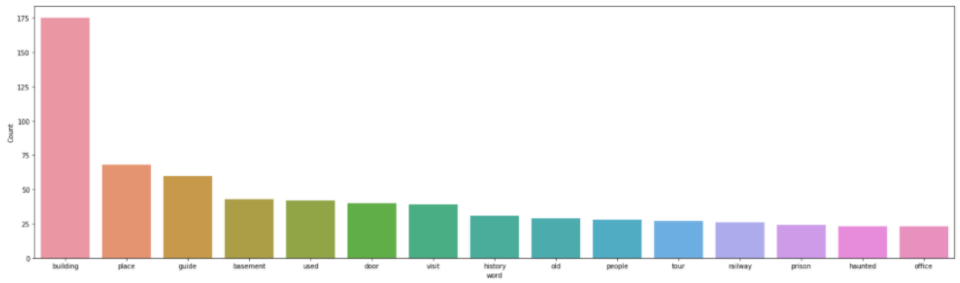
```
179
```

In [54]:
```
# membuat potongan kata (token)

from nltk.tokenize import word_tokenize
tokens_neg = word_tokenize(text_neg)
```

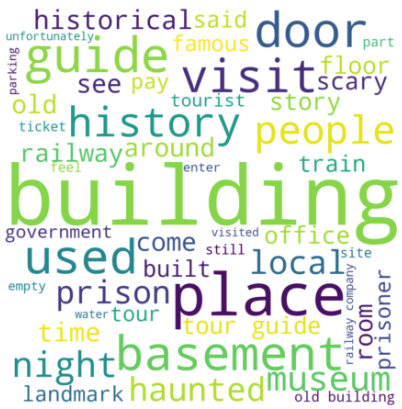
In [55]:
```
# filter out stop words

from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
negative = [w for w in tokens_neg if not w in stop_words]
extend = 'a'
negative = [w for w in negative if not w in extend]

#negative = [nltk.stem.PorterStemmer().stem(word) for word in negative]
negative = [nltk.stem.WordNetLemmatizer().lemmatize(word) for word in negative]
```

In [56]:
```
import string
table = str.maketrans('', '', string.punctuation)
strippedneg = [w.translate(table) for w in negative]
```

In [57]:
```
# memplot kata - kata yang paling sering muncul

freq_words(strippedneg, 15)
```



In [72]:
```
# wordcloud negatif

from wordcloud import WordCloud
from wordcloud import ImageColorGenerator

all_text_negative = ' '.join(str(word) for word in strippedneg)
wordcloud = WordCloud(max_font_size=260, max_words=50, width=1000, height=1000, mode='RGBA', background_color='white').generate(a
plt.figure(figsize=(15,8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.margins(x=0, y=0)
plt.show()
```



Sentimen Positif

In [73]:
```
LawangSewuPos = classificationLawangSewu.loc[classificationLawangSewu['label']==1]
LawangSewuPos.head()
```

Out[73]:

	review	label
0	amazing building collection buildings well res...	1
1	want come hassle free use public transport tax...	1
2	interesting experience historic place place ki...	1
3	place previously known haunted house already r...	1
4	good landmark suggest visit night close	1

In [74]:
```
import difflib

#body_list = df['body'].tolist()
pos_text_list =LawangSewuPos['review'].tolist()

#body = body_list
reviews_pos = pos_text_list
```

In [75]:
```
# stopword

from nltk.corpus import stopwords
english_stopwords = stopwords.words('english')
print(len(english_stopwords))
text_pos = str(pos_text_list)
```

```
179
```

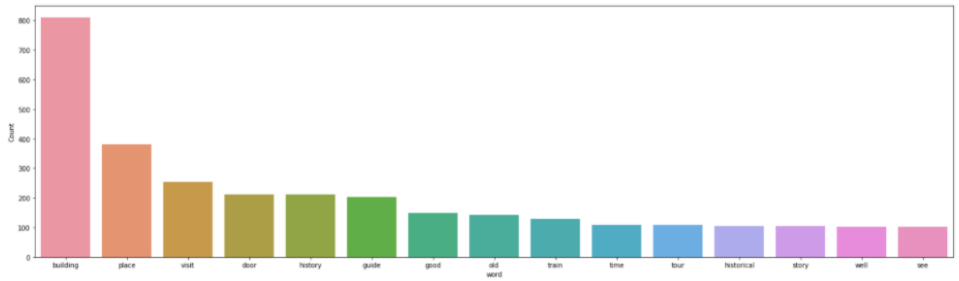
In [76]:
```
# membuat potongan kata (token)

from nltk.tokenize import word_tokenize
tokens_pos = word_tokenize(text_pos)
```

In [77]:
```
# filter out stopwords

from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
positive = [w for w in tokens_pos if not w in stop_words]
positive = [w for w in positive if not w in extend]
```

In [78]:
```
positive = [nltk.stem.WordNetLemmatizer().lemmatize(word) for word in positive]
```

In [79]:
```
import string
table = str.maketrans('', '', string.punctuation)
strippedpos = [w.translate(table) for w in positive]
```

In [80]:
```
# memplot kata - kata yang paling sering muncul

freq_words(strippedpos, 15)
```



In [81]:
```
# wordcloud positif

from wordcloud import WordCloud
from wordcloud import ImageColorGenerator

all_text_positive = ' '.join(str(word) for word in strippedpos)
wordcloud = WordCloud(max_font_size=260, max_words=50, width=1000, height=1000, mode='RGBA', background_color='white').generate(a
plt.figure(figsize=(15,8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.margins(x=0, y=0)
plt.show()
```


museum
tourist
time
interesting
guide
well
still
visit
area
great
love
clean
picture
best
photo
night
tour
renovated
good
place
story
people
tour guide
see
office
nice
visit place
old
way
room
historical
visited
feel
went
building
enjoy
architecture
spooky
used
come
basement
history
train
door
look
better
historical building
around
beautiful
old building

Lampiran 4. *Script* Analisis Data Berbahasa Indonesia

Load data

Dataset yang digunakan adalah data ulasan pengunjung objek wisata Lawang Sewu Kota Semarang berbahasa Indonesia dari situs TripAdvisor yang tercatat sejak Januari 2011 hingga Maret 2020. Pengumpulan data tersebut dilakukan dengan cara web scraping

In [1]:
```
# load dataset kedalam dataframe

import pandas as pd
import numpy as np

df = pd.read_csv('C:/Users/ACER/Tugas Akhir/Scrap Lawang Sewu Indonesia.csv')
print(df.shape)
```

```
(1303, 3)
```

In [2]:
```
# melihat data

df.head()
```

Out[2]:

	penilaian	nama	ulasan
0	5	Ignatius M	Rame dengan pengunjung, letaknya dekat kemana ...
1	5	Deddy	Cocok untuk wisata edukasi sejarah. kalau meng...
2	5	Rizenda	Sangat bagus untuk pembelajaran dan wisata.. k...
3	5	Just Mira	Bagus kl datang pas malem, cakep bt foto2 cuma...
4	5	Kurniadi	Bagus Sekali. Saya berkunjung kamis malam (mal...

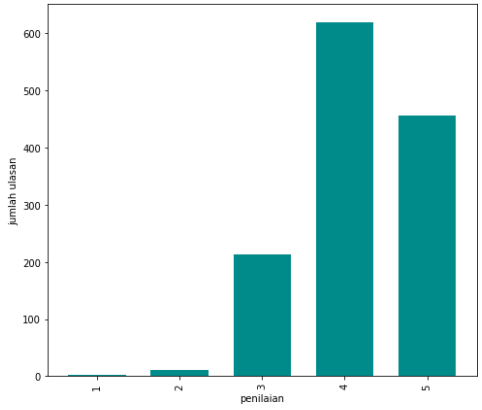
ANALISIS DESKRIPTIF

Melihat informasi secara umum tentang penilaian data ulasan pengunjung objek wisata Lawang Sewu Kota Semarang

In [3]:
```
# melihat banyak ulasan dari tiap penilaian

df['penilaian'].value_counts()
```

Out[3]:
```
4    620
5    456
3    213
2     11
1      3
Name: penilaian, dtype: int64
```

In [4]:
```
# melihat distribusi banyak ulasan berdasar penilaian

import matplotlib.pyplot as plt

_, ax1 = plt.subplots(figsize=(7,6))
stars_histogram = df['penilaian'].value_counts().sort_index()
stars_histogram.plot(kind='bar', width=0.7, color='darkcyan')
plt.xlabel('penilaian')
plt.ylabel('jumlah ulasan')

plt.tight_layout()
plt.show()
```



PREPROCESSING

Pada tahap ini tujuannya adalah mengubah dataset supaya dapat diolah pada algoritma Machine Learning.

In [5]:
```
# menghapus baris jika ada data yang kosong

df = df.dropna()
print(df.shape)
```

```
(1303, 3)
```

Case folding

In [6]:
```
# mengubah semua huruf dalam dokumen menjadi huruf kecil (Lower text)

df['ulasan'] = df['ulasan'].str.lower()
df.head()
```

Out[6]:

	penilaian	nama	ulasan
0	5	Ignatius M	rame dengan pengunjung, letaknya dekat kemana ...
1	5	Deddy	cocok untuk wisata edukasi sejarah. kalau meng...
2	5	Rizenda	sangat bagus untuk pembelajaran dan wisata.. k...
3	5	Just Mira	bagus kl datang pas malem, cakep bt foto2 cuma...
4	5	Kurniadi	bagus sekali. saya berkunjung kamis malam (mal...

In [7]:
```
df.to_csv('hasil case folding.csv')
```

Cleansing

In [8]:
```
# pembersihan teks dari tab, new line, back slice, mention, link, hastag, URL

import string
import re #regex library

def remove_ulasan_special(text):
    # menghapus tab, new line, dan back slice
    text = text.replace('\\t'," ").replace('\\n'," ").replace('\\u'," ").replace('\\',"")
    # menghapus non ASCII (emoticon, chinese word, .etc)
    text = text.encode('ascii', 'replace').decode('ascii')
    # menghapus mention, link, hashtag
    text = ' '.join(re.sub("([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)"," ", text).split())
    # menghapus incomplete URL
    return text.replace("http://", " ").replace("https://", " ")

df['ulasan'] = df['ulasan'].apply(remove_ulasan_special)
df.head()
```

Out[8]:

	penilaian	nama	ulasan
0	5	Ignatius M	rame dengan pengunjung, letaknya dekat kemana ...
1	5	Deddy	cocok untuk wisata edukasi sejarah. kalau meng...
2	5	Rizenda	sangat bagus untuk pembelajaran dan wisata.. k...
3	5	Just Mira	bagus kl datang pas malem, cakep bt foto2 cuma...
4	5	Kurniadi	bagus sekali. saya berkunjung kamis malam (mal...

In [9]:
```
df.to_csv('hasil cleansing.csv')
```

Remove number

In [10]:
```
# menghapus angka

def remove_number(text):
    return  re.sub(r"\d+", "", text)

df['ulasan'] = df['ulasan'].apply(remove_number)
df.head()
```

Out[10]:

	penilaian	nama	ulasan
0	5	Ignatius M	rame dengan pengunjung, letaknya dekat kemana ...
1	5	Deddy	cocok untuk wisata edukasi sejarah. kalau meng...
2	5	Rizenda	sangat bagus untuk pembelajaran dan wisata.. k...
3	5	Just Mira	bagus kl datang pas malem, cakep bt foto cuma ...
4	5	Kurniadi	bagus sekali. saya berkunjung kamis malam (mal...

In [11]:
```
df.to_csv('hasil remove number.csv')
```

Remove punctuation

In [12]:
```
# menghapus tanda baca

def remove_punctuation(text):
    return text.translate(str.maketrans("","",string.punctuation))

df['ulasan'] = df['ulasan'].apply(remove_punctuation)
df.head()
```

Out[12]:

	penilaian	nama	ulasan
0	5	Ignatius M	rame dengan pengunjung letaknya dekat kemana ...
1	5	Deddy	cocok untuk wisata edukasi sejarah kalau mengu...
2	5	Rizenda	sangat bagus untuk pembelajaran dan wisata kur...
3	5	Just Mira	bagus kl datang pas malem cakep bt foto cuma r...
4	5	Kurniadi	bagus sekali saya berkunjung kamis malam malam...

In [13]:
```
df.to_csv('hasil remove punctuation.csv')
```

Remove single char

In [14]:
```
# menghapus huruf tunggal (tidak berbentuk kata)

def remove_singl_char(text):
    return re.sub(r"\b[a-zA-Z]\b", "", text)

df['ulasan'] = df['ulasan'].apply(remove_singl_char)
df.head()
```

Out[14]:

	penilaian	nama	ulasan
0	5	Ignatius M	rame dengan pengunjung letaknya dekat kemana ...
1	5	Deddy	cocok untuk wisata edukasi sejarah kalau mengu...
2	5	Rizenda	sangat bagus untuk pembelajaran dan wisata kur...
3	5	Just Mira	bagus kl datang pas malem cakep bt foto cuma r...
4	5	Kurniadi	bagus sekali saya berkunjung kamis malam malam...

In [15]:
```
df.to_csv('hasil remove single char.csv')
```

Tokenizing

In [16]:
```
# pemisahan teks menjadi potongan-potongan kata yang disebut sebagai token

from nltk.tokenize import word_tokenize

# NLTK word tokenize
def word_tokenize_wrapper(text):
    return word_tokenize(text)

df['ulasan'] = df['ulasan'].apply(word_tokenize_wrapper)
df.head()
```

Out[16]:

	penilaian	nama	ulasan
0	5	Ignatius M	[rame, dengan, pengunjung, letaknya, dekat, ke...
1	5	Deddy	[cocok, untuk, wisata, edukasi, sejarah, kalau...
2	5	Rizenda	[sangat, bagus, untuk, pembelajaran, dan, wisa...
3	5	Just Mira	[bagus, kl, datang, pas, malem, cakep, bt, fot...
4	5	Kurniadi	[bagus, sekali, saya, berkunjung, kamis, malam...

In [17]:
```
df.to_csv('hasil tokenizing.csv')
```

Spell normalization

In [18]:
```
# menyeragamkan kata yang memiliki makna yang sama namun penulisannya berbeda

normalizad_word = pd.read_excel("normalisasi.xlsx")
normalizad_word_dict = {}

for index, row in normalizad_word.iterrows():
    if row[0] not in normalizad_word_dict:
        normalizad_word_dict[row[0]] = row[1]

def normalized_term(document):
    return [normalizad_word_dict[term] if term in normalizad_word_dict else term for term in document]

df['ulasan'] = df['ulasan'].apply(normalized_term)
df.head()
```

Out[18]:

	penilaian	nama	ulasan
0	5	Ignatius M	[ramai, dengan, pengunjung, letaknya, dekat, k...
1	5	Deddy	[cocok, untuk, wisata, edukasi, sejarah, kalau...
2	5	Rizenda	[sangat, bagus, untuk, pembelajaran, dan, wisa...
3	5	Just Mira	[bagus, kalau, datang, pas, malem, cakep, buat...
4	5	Kurniadi	[bagus, sekali, saya, berkunjung, kamis, malam...

In [19]:
```
df.to_csv('hasil spell normalization.csv')
```

Filtering

In [20]:
```
# memfilter untuk mengambil kata-kata penting dengan menggunakan algoritma stoplist (membuang kata kurang penting)
# stopword adalah kata umum yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna
# contoh stopword dalam bahasa Indonesia adalah “yang”, “dan”, “di”, “dari”, "dengan", "untuk" dll.

# import stopword from NLTK
from nltk.corpus import stopwords

# dapatkan stopword indonesia dari NLTK stopword
list_stopwords = stopwords.words('indonesian')

# menambahkan stopword tambahan
list_stopwords.extend(['pas', 'ya', 'sih', 'deh', 'loh', 'oiya', 'nih', 'ok', 'ah'])
```

```
# menambahkan stopword dari file csv
txt_stopword = pd.read_csv('C:/Users/ACER/Tugas Akhir/stopword indonesia.txt', names= ["stopwords"], header = None)

# mengkonversi kata stopword tambahan & stopword pada file csv ke list
list_stopwords.extend(txt_stopword["stopwords"][0].split(' '))

# mengkonversi list ke dictionary
list_stopwords = set(list_stopwords)

# menghapus stopword pada data
def stopwords_removal(words):
    return [word for word in words if word not in list_stopwords]

df['ulasan'] = df['ulasan'].apply(stopwords_removal)
df.head()
```

Out[20]:

	penilaian	nama	ulasan
0	5	Ignatius M	[ramai, pengunjung, letaknya, kemana, disekita...
1	5	Deddy	[cocok, wisata, edukasi, sejarah, mengunjungi,...
2	5	Rizenda	[bagus, pembelajaran, wisata, leluasa, mengeks...
3	5	Just Mira	[bagus, malem, cakep, foto, ngeri, sedap, angk...
4	5	Kurniadi	[bagus, berkunjung, kamis, malam, malam, jumat...

In [21]:
```
df.to_csv('hasil filtering.csv')
```

In [22]:
```
df.to_csv('hasil preprocessing indonesia.csv')
```

In [23]:
```
# load data hasil preprocessing kedalam dataframe

import pandas as pd
import numpy as np

df = pd.read_csv('C:/Users/Acer/Tugas Akhir/hasil preprocessing indonesia.csv', usecols=['penilaian', 'nama', 'ulasan'])
df.columns = ['penilaian', 'nama', 'ulasan']

df.head()
```

Out[23]:

	penilaian	nama	ulasan
0	5	Ignatius M	['ramai', 'pengunjung', 'letaknya', 'kemana', ...
1	5	Deddy	['cocok', 'wisata', 'edukasi', 'sejarah', 'men...
2	5	Rizenda	['bagus', 'pembelajaran', 'wisata', 'leluasa',...
3	5	Just Mira	['bagus', 'malem', 'cakep', 'foto', 'ngeri', '...
4	5	Kurniadi	['bagus', 'berkunjung', 'kamis', 'malam', 'mal...

In [24]:
```
# menggabungkan daftar token menjadi dokumen string tunggal

import ast

def join_text_list(texts):
    texts = ast.literal_eval(texts)
    return ' '.join([text for text in texts])

df['ulasan'] = df['ulasan'].apply(join_text_list)
df.head()
```

Out[24]:

	penilaian	nama	ulasan
0	5	Ignatius M	ramai pengunjung letaknya kemana disekitaran p...
1	5	Deddy	cocok wisata edukasi sejarah mengunjungi sejar...
2	5	Rizenda	bagus pembelajaran wisata leluasa mengeksplor ...
3	5	Just Mira	bagus malem cakep foto ngeri sedap angker bang...
4	5	Kurniadi	bagus berkunjung kamis malam malam jumat senga...

In [25]:
```
df.to_csv('hasil Indonesia.csv')
```

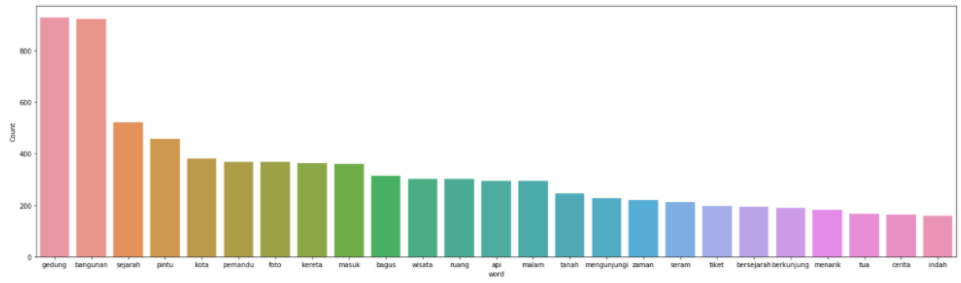
In [26]:
```
import difflib

#body_list = df['body'].tolist()
review_text_list = df['ulasan'].tolist()

#body = body_list
reviews = review_text_list
s = difflib.SequenceMatcher(None, reviews).ratio()
print ("ratio:", s, "\n")
```

```
ratio: 0.0
```

Frekuensi & TFIDF

In [27]:
```
# fungsi untuk memplot term yang paling sering muncul

from nltk import FreqDist
import seaborn as sns

def freq_words(x, terms = 30):
  all_words = ' '.join([text for text in x])
  all_words = all_words.split()

  fdist = FreqDist(all_words)
  words_df = pd.DataFrame({'word':list(fdist.keys()), 'count':list(fdist.values())})

  # memilih 25 term yang paling sering muncul
  d = words_df.nlargest(columns="count", n = terms)
  plt.figure(figsize=(25,7))
  ax = sns.barplot(data=d, x= "word", y = "count")
  ax.set(ylabel = 'Count')
  plt.show()
```

In [28]:
```
# plot term yang paling sering

freq_words(df['ulasan'], 25)
```



In [29]:
```
# pembobotan kata

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
cvec = CountVectorizer(stop_words='english', min_df=1, max_df=.5, ngram_range=(1,2))
cvec
```

Out[29]:
```
CountVectorizer(max_df=0.5, ngram_range=(1, 2), stop_words='english')
```

In [30]:
```
# menghitung semua n-gram yang ditemukan di semua dokumen

from itertools import islice
from nltk.corpus import stopwords
global str

cvec.fit(review_text_list)
list(islice(cvec.vocabulary_.items(), 20))

len(cvec.vocabulary_)
```

Out[30]:
```
23201
```

In [31]:
```
cvec = CountVectorizer(stop_words='english', min_df=.0025, max_df=.5, ngram_range=(1,2))
cvec.fit(review_text_list)
len(cvec.vocabulary_)
```

Out[31]:
```
1722
```

In [32]:
```
cvec_counts = cvec.transform(review_text_list)
print('sparse matrix shape:', cvec_counts.shape)
print('nonzero count:', cvec_counts.nnz)
print('sparsity: %.2f%%' % (100.0 * cvec_counts.nnz / (cvec_counts.shape[0] * cvec_counts.shape[1])))
```

```
sparse matrix shape: (1303, 1722)
nonzero count: 27598
sparsity: 1.23%
```

In [33]:
```
# menghitung frekuensi kemunculan term

occ = np.asarray(cvec_counts.sum(axis=0)).ravel().tolist()
counts_df = pd.DataFrame({'term': cvec.get_feature_names(), 'occurrences': occ})
counts_df.sort_values(by='occurrences', ascending=False).head(20)
```

Out[33]:

	term	occurrences
511	gedung	927
105	bangunan	924
1421	sejarah	521
1315	pintu	457
758	kota	381
1205	pemandu	369

In [34]:
```
# Sekarang kita memiliki jumlah term untuk setiap dokumen kita dapat menggunakan Tfidf Transformer untuk menghitung
# Bobot untuk setiap istilah dalam setiap dokumen

transformer = TfidfTransformer()
transformed_weights = transformer.fit_transform(cvec_counts)
transformed_weights
```

Out[34]:
```
<1303x1722 sparse matrix of type '<class 'numpy.float64'>'
        with 27598 stored elements in Compressed Sparse Row format>
```

In [35]:
```
# melihat 20 term teratas dengan weight rata-rata tf-idf

weights = np.asarray(transformed_weights.mean(axis=0)).ravel().tolist()
weights_df = pd.DataFrame({'term': cvec.get_feature_names(), 'weight': weights})
weights_df.sort_values(by='weight', ascending=False).head(20)
```

Out[35]:

	term	weight
105	bangunan	0.055212
511	gedung	0.053636
1421	sejarah	0.038385
1315	pintu	0.033533
758	kota	0.032356
81	bagus	0.031646

PELABELAN

In [36]:
```
# membuat kamus kata

word_dict = {}
for i in range(0,len(df['ulasan'])):
    sentence = df['ulasan'][i]
    word_token = word_tokenize(sentence)
    for j in word_token:
        if j not in word_dict:
            word_dict[j] = 1
        else:
            word_dict[j] += 1
```

In [37]:
```
len(word_dict)
```

Out[37]:
```
4157
```

In [38]:
```
len({k:v for (k,v) in word_dict.items() if v < 4})
```

Out[38]:
```
3113
```

In [39]:
```
# import lexicon, dan hapus kata-kata negasi dari leksikon

negasi = ['bukan','tidak','ga','gk']
lexicon = pd.read_csv('C:/Users/ACER/Tugas Akhir/modified_full_lexicon.csv')
lexicon = lexicon.drop(lexicon[(lexicon['word'] == 'bukan')
                               |(lexicon['word'] == 'tidak')
                               |(lexicon['word'] == 'ga')|(lexicon['word'] == 'gk') ].index,axis=0)
lexicon = lexicon.reset_index(drop=True)
```

In [40]:
```
len(lexicon)
```

Out[40]:
```
10249
```

In [41]:
```
lexicon.head(10)
```

Out[41]:

	word	weight	number_of_words
0	hai	3	1
1	merekam	2	1
2	ekstensif	3	1
3	paripurna	1	1
4	detail	2	1

In [42]:
```
lexicon_word = lexicon['word'].to_list()
lexicon_num_words = lexicon['number_of_words']
```

In [43]:
```
len(lexicon_word)
```

Out[43]:
```
10249
```

In [44]:
```
# Memeriksa apakah ada kata dalam kamus yang tidak termasuk dalam lexicon

from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

ns_words = []
factory = StemmerFactory()
stemmer = factory.create_stemmer()
for word in word_dict.keys():
    if word not in lexicon_word:
        kata_dasar = stemmer.stem(word)
        if kata_dasar not in lexicon_word:
            ns_words.append(word)
len(ns_words)
```

Out[44]:
```
2130
```

In [45]:
```
# melihat jenis kata, dimulai dengan beberapa kata yang memiliki banyak kemunculan karena kemungkinan besar ini bukan tipe case

len({k:v for (k,v) in word_dict.items() if ((k in ns_words)&(v>3)) })
```

Out[45]: 436

In [46]:
```
ns_words_list = {k:v for (k,v) in word_dict.items() if ((k in ns_words)&(v>3))}
```

In [47]:
```
# Ternyata kata-kata yang tidak termasuk dalam lexicon, adalah kata yang tidak memiliki arti sentimen

sort_orders = sorted(ns_words_list.items(), key=lambda x: x[1], reverse=True)
sort_orders=sort_orders[0:20]
for i in sort_orders:

    print(i[0], i[1])
```

```
gedung 927
bangunan 924
sejarah 521
pintu 457
kota 381
```

In [48]:
```
# sentiment

lexicon['number_of_words'].value_counts()
```

Out[48]:
```
1    9536
2     687
3      24
4       2
Name: number_of_words, dtype: int64
```

In [49]:
```
'pekerti' in word_dict
```

Out[49]: False

In [50]:
```
'budi baik' in lexicon_word
```

Out[50]: True

In [51]:
```
# menghitung sentimen kata dengan menghitungnya menjadi lexicon
sencol =[]
senrow =np.array([])
nsen = 0
factory = StemmerFactory()
stemmer = factory.create_stemmer()
sentiment_list = []

# fungsi untuk menuliskan sentimen kata jika sudah ditemukan
def found_word(ind,words,word,sen,sencol,sentiment,add):
    # jika sudah termasuk dalam bag of words matrix, maka naikkan (tambahkan) saja nilainya
    if word in sencol:
        sen[sencol.index(word)] += 1
    else:
    # jika tidak, maka tambahkan kata baru
        sencol.append(word)
        sen.append(1)
        add += 1
    # jika ada kata negasi sebelumnya, sentimennya adalah negasi sentimennya
    if (words[ind-1] in negasi):
        sentiment += -lexicon['weight'][lexicon_word.index(word)]
    else:
        sentiment += lexicon['weight'][lexicon_word.index(word)]

    return sen,sencol,sentiment,add
# memeriksa setiap kata, jika muncul dalam lexicon, dan kemudian menghitung sentimennya
for i in range(len(df)):
    nsen = senrow.shape[0]
    words = word_tokenize(df['ulasan'][i])
    sentiment = 0
    add = 0
    prev = [0 for ii in range(len(words))]
    n_words = len(words)
    if len(sencol)>0:
        sen =[0 for j in range(len(sencol))]
    else:
        sen =[]

    for word in words:
        ind = words.index(word)
        # periksa apakah mereka termasuk dalam lexicon
        if word in lexicon_word :
            sen,sencol,sentiment,add= found_word(ind,words,word,sen,sencol,sentiment,add)
        else:
        # jika tidak, maka periksa kata dasarnya
            kata_dasar = stemmer.stem(word)
            if kata_dasar in lexicon_word:
                sen,sencol,sentiment,add= found_word(ind,words,kata_dasar,sen,sencol,sentiment,add)
        # jika masih negatif, coba cocokkan kombinasi kata dengan kata yang berdekatan
            elif(n_words>1):
                if ind-1>-1:
                    back_1    = words[ind-1]+' '+word
                    if (back_1 in lexicon_word):
                        sen,sencol,sentiment,add= found_word(ind,words,back_1,sen,sencol,sentiment,add)
                    elif(ind-2>-1):
                        back_2    = words[ind-2]+' '+back_1
                        if back_2 in lexicon_word:
                            sen,sencol,sentiment,add= found_word(ind,words,back_2,sen,sencol,sentiment,add)
```

```
    # jika ada kata baru yang ditemukan, maka perluas matriks
    if add>0:
        if i>0:
            if (nsen==0):
                senrow = np.zeros([i,add],dtype=int)
            elif(i!=nsen):
                padding_h = np.zeros([nsen,add],dtype=int)
                senrow = np.hstack((senrow,padding_h))
                padding_v = np.zeros([(i-nsen),senrow.shape[1]],dtype=int)
                senrow = np.vstack((senrow,padding_v))
            else:
                padding =np.zeros([nsen,add],dtype=int)
                senrow = np.hstack((senrow,padding))
            senrow = np.vstack((senrow,sen))
        if i==0:
            senrow = np.array(sen).reshape(1,len(sen))
    # jika tidak ada maka perbarui saja matriks lama
    elif(nsen>0):
        senrow = np.vstack((senrow,sen))

    sentiment_list.append(sentiment)
```

In [52]:
```
# membangun kerangka data yang berisi sekumpulan kata dan sentimen yang telah dihitung sebelumnya

sencol.append('sentiment')
sentiment_array = np.array(sentiment_list).reshape(senrow.shape[0],1)
sentiment_data = np.hstack((senrow,sentiment_array))
df_sen = pd.DataFrame(sentiment_data,columns = sencol)
```

In [53]:
```
df_sen.head(5)
```

Out[53]:

	ramai	pilihan	hotel	cocok	anak	bagus	ajar	ruang	tersedia	malem	...	marah	menghapus	acu	rendah	latihan	memeriksa	menetap	selebaran	ju
0	1	1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
1	0	0	0	1	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	1	1	1	1	0	...	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	1	0	0	0	1	...	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	2	0	0	0	0	...	0	0	0	0	0	0	0	0	

5 rows × 1352 columns

In [54]:
```
# melihat sentimen dari data original

res_df = pd.DataFrame([])
res_df['penilaian'] = df['penilaian'].copy()
res_df['nama'] = df['nama'].copy()
res_df['ulasan'] = df['ulasan'].copy()
res_df['sentimen']  = df_sen['sentiment'].copy()
```

In [55]:
```
res_df.head()
```

Out[55]:

	penilaian	nama	ulasan	sentimen
0	5	Ignatius M	ramai pengunjung letaknya kemana disekitaran p...	9
1	5	Deddy	cocok wisata edukasi sejarah mengunjungi sejar...	1
2	5	Rizenda	bagus pembelajaran wisata leluasa mengeksplor ...	4
3	5	Just Mira	bagus malem cakep foto ngeri sedap angker bang...	8
4	5	Kurniadi	bagus berkunjung kamis malam malam jumat senga...	22

In [56]:
```
# mengubah penilain menjadi label
# pelabelan data, data akan berlabel negatif(0) jika nilai compound <0,0 dan akan berlabel positif(1) jika nilai compound >=0,0

label = []
for index, row in res_df.iterrows():
    if row["sentimen"] >= 0:
        label.append(1)
    else:
        label.append(0)

res_df["label"] = label
res_df = res_df.drop(columns=['sentimen'])
res_df.head()
```

Out[56]:

	penilaian	nama	ulasan	label
0	5	Ignatius M	ramai pengunjung letaknya kemana disekitaran p...	1
1	5	Deddy	cocok wisata edukasi sejarah mengunjungi sejar...	1
2	5	Rizenda	bagus pembelajaran wisata leluasa mengeksplor ...	1
3	5	Just Mira	bagus malem cakep foto ngeri sedap angker bang...	1
4	5	Kurniadi	bagus berkunjung kamis malam malam jumat senga...	1

In [57]:
```
# melihat banyak ulasan dari tiap label

res_df['label'].value_counts()
```

Out[57]:
```
1    1017
0     286
Name: label, dtype: int64
```

METODE KLASIFIKASI

In [70]:
```
klasifikasiLawangSewu = res_df
klasifikasiLawangSewu = res_df.drop(columns=['penilaian', 'nama'])
klasifikasiLawangSewu.head()
```

Out[70]:

	ulasan	label
0	ramai pengunjung letaknya kemana disekitaran p...	1
1	cocok wisata edukasi sejarah mengunjungi sejar...	1
2	bagus pembelajaran wisata leluasa mengeksplor ...	1
3	bagus malem cakep foto ngeri sedap angker bang...	1
4	bagus berkunjung kamis malam malam jumat senga...	1

In [71]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import model_selection, svm
from sklearn.metrics import accuracy_score
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder
```

In [72]:
```
# data training & data testing

Train_X, Test_X, Train_Y, Test_Y = model_selection.train_test_split(klasifikasiLawangSewu['ulasan'],klasifikasiLawangSewu['label'

Encoder = LabelEncoder()
Train_Y = Encoder.fit_transform(Train_Y)
Test_Y = Encoder.fit_transform(Test_Y)

Tfidf_vect = TfidfVectorizer(max_features=5000)
Tfidf_vect.fit(klasifikasiLawangSewu['ulasan'])
Train_X_Tfidf = Tfidf_vect.transform(Train_X)
Test_X_Tfidf = Tfidf_vect.transform(Test_X)
```

In [73]:
```
len(Train_X)
```

Out[73]: 1042

In [74]:
```
len(Test_X)
```

Out[74]: 261

Support Vector Mechine

In [75]:
```
# Support Vector Mechine

SVM = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='scale')
SVM.fit(Train_X_Tfidf,Train_Y)

predictions_SVM = SVM.predict(Test_X_Tfidf)

print("SVM Accuracy Score -> ",accuracy_score(predictions_SVM, Test_Y)*100)
```

```
SVM Accuracy Score ->  86.20689655172413
```

Naive Bayes

In [76]:
```
# Naive Bayes

from sklearn import naive_bayes
NB = naive_bayes.MultinomialNB()
NB.fit(Train_X_Tfidf,Train_Y)

predictions_NB = NB.predict(Test_X_Tfidf)

print("NB Accuracy Score -> ",accuracy_score(predictions_NB, Test_Y)*100)
```

```
NB Accuracy Score ->  78.9272030651341
```

Random Forest

In [77]:
```
# Random Forest

from sklearn.ensemble import RandomForestClassifier
RF = RandomForestClassifier()
RF.fit(Train_X_Tfidf,Train_Y)

predictions_RF = RF.predict(Test_X_Tfidf)

print("RF Accuracy Score -> ",accuracy_score(predictions_RF, Test_Y)*100)
```

```
RF Accuracy Score ->  80.07662835249042
```

Decision Tree

In [87]:
```
# Decision Tree

from sklearn.tree import DecisionTreeClassifier
DT = DecisionTreeClassifier()
DT.fit(Train_X_Tfidf,Train_Y)

predictions_DT = DT.predict(Test_X_Tfidf)

print("DT Accuracy Score -> ",accuracy_score(predictions_DT, Test_Y)*100)
```

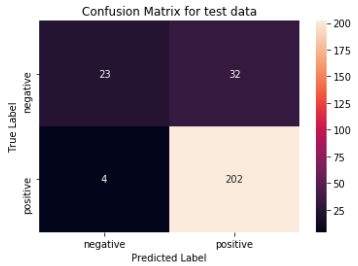
```
DT Accuracy Score ->  72.41379310344827
```

Logistic Regression

In [79]:
```
# Logistic Regression

from sklearn.linear_model import LogisticRegression
LR = LogisticRegression()
LR.fit(Train_X_Tfidf,Train_Y)

predictions_LR = LR.predict(Test_X_Tfidf)

print("LR Accuracy Score -> ",accuracy_score(predictions_LR, Test_Y)*100)
```

```
LR Accuracy Score ->  80.45977011494253
```

In [80]:
```
# confusion matrix

from sklearn.metrics import confusion_matrix
import seaborn as sns

conf_mat = confusion_matrix(Test_Y, predictions_SVM)
class_label = ["negative", "positive"]
test = pd.DataFrame(conf_mat, index = class_label, columns = class_label)
sns.heatmap(test, annot = True,fmt="d")
plt.title("Confusion Matrix for test data")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.show()
```



VISUALISASI

Sentimen negatif

In [69]:
```
LawangSewuNeg = klasifikasiLawangSewu.loc[klasifikasiLawangSewu['label']== 0]
LawangSewuNeg.head()
```

Out[69]:

	ulasan	label
9	lumayan ngeluarin kocek yak mitosnya percaya a...	0
13	kesemarang ketemu teman disana perjalanan kosa...	0
18	masyarakat menyebut bangunan tua bersejarah pe...	0
19	wisata unik mencekam disana hawa lumayan angke...	0
21	ketempat jam sore menghabiskan senja sembari d...	0

In [71]:
```
import difflib

#body_list = df['body'].tolist()
neg_text_list = LawangSewuNeg['ulasan'].tolist()

#body = body_list
reviews_neg = neg_text_list
```

In [72]:
```
# stopword

from nltk.corpus import stopwords
english_stopwords = stopwords.words('english')
print(len(english_stopwords))
text_neg = str(neg_text_list)
```

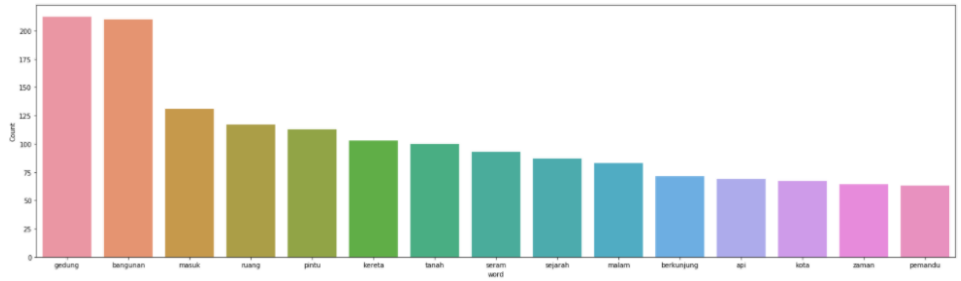
```
179
```

In [73]:
```
# membuat potongan kata (token)

from nltk.tokenize import word_tokenize
tokens_neg = word_tokenize(text_neg)
```

In [74]:
```
# filter out stop words

import nltk
from nltk.corpus import stopwords

stop_words = set(stopwords.words('english'))
negative = [w for w in tokens_neg if not w in stop_words]
extend = 'a'
negative = [w for w in negative if not w in extend]

#negative = [nltk.stem.PorterStemmer().stem(word) for word in negative]
negative = [nltk.stem.WordNetLemmatizer().lemmatize(word) for word in negative]
```

In [75]:
```
import string
table = str.maketrans('', '', string.punctuation)
strippedneg = [w.translate(table) for w in negative]
```

In [76]:
```
# memplot kata - kata yang paling sering muncul

freq_words(strippedneg, 15)
```



In [77]:
```
# wordcloud negatif

from wordcloud import WordCloud
from wordcloud import ImageColorGenerator

all_text_negative = ' '.join(str(word) for word in strippedneg)
wordcloud = WordCloud(max_font_size=260, max_words=50, width=700, height=700, mode='RGBA', background_color='white').generate(all
plt.figure(figsize=(15,8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.margins(x=0, y=0)
plt.show()
```

Sentimen positif

In [78]:
```
LawangSewuPos = klasifikasiLawangSewu.loc[klasifikasiLawangSewu['label']==1]
LawangSewuPos.head()
```

Out[78]:

	ulasan	label
0	ramai pengunjung letaknya kemana disekitaran p...	1
1	cocok wisata edukasi sejarah mengunjungi sejar...	1
2	bagus pembelajaran wisata leluasa mengeksplor ...	1
3	bagus malem cakep foto ngeri sedap angker bang...	1
4	bagus berkunjung kamis malam malam jumat senga...	1

In [79]:
```
import difflib

#body_list = df['body'].tolist()
pos_text_list =LawangSewuPos['ulasan'].tolist()

#body = body_list
reviews_pos = pos_text_list
```

In [80]:
```
# stopword

from nltk.corpus import stopwords
english_stopwords = stopwords.words('english')
print(len(english_stopwords))
text_pos = str(pos_text_list)
```

```
179
```

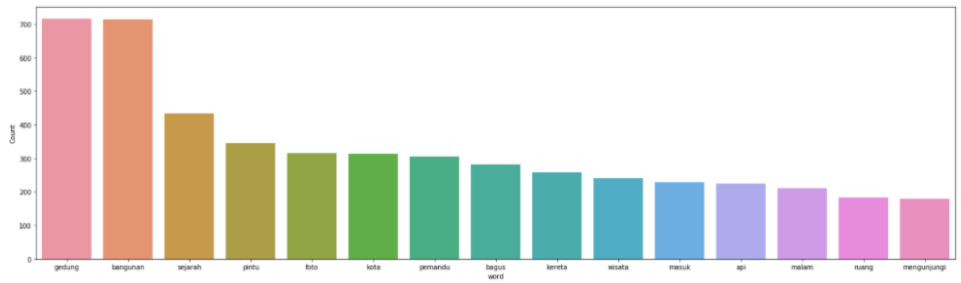
In [81]:
```
# membuat potongan kata (token)

from nltk.tokenize import word_tokenize
tokens_pos = word_tokenize(text_pos)
```

In [82]:
```
# filter out stopwords

from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
positive = [w for w in tokens_pos if not w in stop_words]
positive = [w for w in positive if not w in extend]
```

In [83]:
```
positive = [nltk.stem.WordNetLemmatizer().lemmatize(word) for word in positive]
```

In [84]:
```
import string
table = str.maketrans('', '', string.punctuation)
strippedpos = [w.translate(table) for w in positive]
```

In [85]:
```
# memplot kata - kata yang paling sering muncul

freq_words(strippedpos, 15)
```



In [89]:
```
# wordcloud positif

from wordcloud import WordCloud
from wordcloud import ImageColorGenerator

# positive
all_text_positive = ' '.join(str(word) for word in strippedpos)
wordcloud = WordCloud(max_font_size=260, max_words=50, width=1000, height=1000, mode='RGBA', background_color='white').generate(a
plt.figure(figsize=(15,8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.margins(x=0, y=0)
plt.show()
```


suasana
arsitektur
pemandu
dikunjungi
parkir
sejarah
wisata
seram
foto
siang
berfoto
menarik
suka
renovasi
direnovasi
jalan
bangunan
cerita
unik
bangunan bersejarah
bersih
malam
bagus
indah
ramai
terkenal
menikmati
kereta
api
zaman
sayang
tanah
tiket masuk
bangunannya
berkunjung
mengunjungi
museum
bersejarah
pengunjung
gedung
pusat kota
perjalanan
pergi
lantai
ruang
pintu
kota
mistis
terawat
masuk
tur pemandu