TUGAS AKHIR – KS184822

KLASIFIKASI SENTIMEN WISATAWAN CANDI BOROBUDUR PADA SITUS *TRIPADVISOR* MENGGUNAKAN *SUPPORT VECTOR MACHINE* DAN *K-NEAREST NEIGHBOR*

RAHAYU PRIHATINI SAPUTRI
NRP 062115 4000 0040

Dosen Pembimbing
Dra. Wiwiek Setya Winahju, M.S.
Dr. Dra. Kartika Fithriasari, M.Si.

PROGRAM STUDI SARJANA
DEPARTEMEN STATISTIKA
FAKULTAS MATEMATIKA, KOMPUTASI, DAN SAINS DATA
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2019

TUGAS AKHIR – KS184822

KLASIFIKASI SENTIMEN WISATAWAN CANDI BOROBUDUR PADA SITUS TRIPADVISOR MENGGUNAKAN *K-NEAREST NEIGHBOR* DAN *SUPPORT VECTOR MACHINE*

RAHAYU PRIHATINI SAPUTRI
NRP 062115 4000 0040

Dosen Pembimbing
Dra. Wiwiek Setya Winahju, M.S.
Dr. Dra. Kartika Fithriasari, M.Si.

PROGRAM STUDI SARJANA
DEPARTEMEN STATISTIKA
FAKULTAS MATEMATIKA, KOMPUTASI, DAN SAINS DATA
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2019

FINAL PROJECT – KS184822

SENTIMENT CLASSIFICATION OF BOROBUDUR TEMPLE VISITORS REVIEW ON TRIPADVISOR SITE USING K-NEAREST NEIGHBOR AND SUPPORT VECTOR MACHINE

RAHAYU PRIHATINI SAPUTRI
SN 062114 4000 0040

Supervisors
Dra. Wiwiek Setya Winahju, M.S.
Dr. Dra. Kartika Fithriasari, M.Si.

UNDERGRADUATE PROGRAMME
DEPARTMENT OF STATISTICS
FACULTY OF MATHEMATICS, COMPUTING, AND DATA SCIENCE
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2019

LEMBAR PENGESAHAN

KLASIFIKASI SENTIMEN WISATAWAN CANDI BOROBUDUR PADA SITUS *TRIPADVISOR* MENGGUNAKAN *K-NEAREST NEIGHBOR* DAN *SUPPORT VECTOR MACHINE*

TUGAS AKHIR

Diajukan untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Statistika
pada
Program Studi Sarjana Departemen Statistika
Fakultas Matematika, Komputasi, dan Sains Data
Institut Teknologi Sepuluh Nopember

Oleh :
Rahayu Prihatini Saputri
NRP. 062115 4000 0040

Disetujui oleh Pembimbing:
Dra. Wiwiek Setya Winahju, M.S. ( )
NIP. 19560424 198303 2 001
Dr. Dra. Kartika Fithriasari, M.Si. ( )
NIP. 19691212 199303 2 002

Mengetahui,
Kepala Departemen Statistika

Dr. Suhartono
NIP. 19710929 199512 1 001

SURABAYA, JULI 2019

(Halaman ini sengaja dikosongkan)

KLASIFIKASI SENTIMEN WISATAWAN CANDI BOROBUDUR PADA SITUS *TRIPADVISOR* MENGGUNAKAN *K-NEAREST NEIGHBOR* DAN *SUPPORT VECTOR MACHINE*

Nama Mahasiswa	**:**	**Rahayu Prihatini Saputri**
NRP	**:**	**062115 4000 0040**
Departemen	**:**	**Statistika**
Dosen Pembimbing	**:**	**Dra. Wiwiek Setya Winahju, M.S.**
		Dr. Dra. Kartika Fithriasari, M.Si.

Abstrak

Candi Borobudur merupakan salah satu destinasi wisata di Indonesia yang saat ini menjadi satu dari sepuluh destinasi yang diprioritaskan oleh Kementerian Pariwisata. Untuk memajukan wisata Candi Borobudur, pihak pengelola wisata perlu mengetahui berbagai persepsi dari wisatawan. Oleh sebab itu diperlukan suatu teknik untuk mendapatkan sentimen wisatawan dari ulasan yang tersedia di situs TripAdvisor dengan metode klasifikasi. Pada penelitian ini digunakan metode K-Nearest Neighbor (K-NN) dan Support Vector Machine (SVM), yang disertai dengan penerapan N-gram sebagai teknik penggabungan beberapa kata berurutan dan Synthetic Minority Oversampling Technique (SMOTE) untuk mengatasi kasus data imbalance. Hasil dari penelitian ini menunjukkan bahwa metode terbaik yang dapat diterapkan untuk mengklasifikasikan sentimen wisatawan Candi Borobudur adalah SVM kernel Radial Basis Function (RBF) yang disertai dengan menerapkan teknik unigram. Kinerja klasifikasi yang dihasilkan oleh metode ini tergolong sangat baik.

Kata kunci: K-Nearest Neighbor, N-gram, Sentimen, Synthetic Minority Oversampling Technique, Support Vector Machine

(Halaman ini sengaja dikosongkan)

SENTIMENT CLASSIFICATION OF BOROBUDUR TEMPLE VISITORS REVIEW ON TRIPADVISOR SITE USING K-NEAREST NEIGHBOR AND SUPPORT VECTOR MACHINE

Name	**:**	**Rahayu Prihatini Saputri**
Student Number	**:**	**062115 4000 0040**
Department	**:**	**Statistics**
Supervisor	**:**	**Dra. Wiwiek Setya Winahju, M.S.**
		Dr. Dra. Kartika Fithriasari, M.Si.

Abstract

Borobudur Temple is one of the destinastions in Indonesia which is currently one of ten destinations prioritized by the Ministry of Tourism. The manager needs to know various perceptions from visitors to advance the destinations. Therefore, a technique is needed to get visitor sentiments from the reviews on the TripAdvisor site with classification method. The methods used in this study are K-Nearest Neighbor (K-NN) and Support Vector Machine (SVM), which are accompanied by the application of N-gram as a technique of combining sequential words and Synthetic Minority Oversampling Technique (SMOTE) to handle imbalanced dataset. The results of this study indicate that the best method that can be applied to classify visitor sentiments of Borobudur Temple is SVM with Radial Basis Function (RBF) kernel and apply the unigram technique. This method shows a very good classification performance.

Keywords: K-Nearest Neighbor, N-gram, Sentiment, Synthetic Minority Oversampling Technique, Support Vector Machine

(Halaman ini sengaja dikosongkan)

KATA PENGANTAR

Puji syukur penulis panjatkan atas rahmat dan hidayah yang diberikan Allah SWT sehingga penulis dapat menyelesaikan laporan Tugas Akhir yang berjudul "Klasifikasi Sentimen Wisatawan Candi Borobudur Pada Situs *Tripadvisor* menggunakan *K-Nearest Neighbor* dan *Support Vector Machine*" dengan lancar.

Penulis menyadari bahwa Tugas Akhir ini dapat terselesaikan tidak terlepas dari bantuan dan dukungan berbagai pihak. Oleh karena itu, penulis menyampaikan terima kasih kepada:

1. Kedua orang tua, atas segala doa, nasihat, kasih sayang, dan dukungan yang diberikan kepada penulis demi kesuksesan penulis.
2. Dr. Suhartono selaku Ketua Departemen Statistika dan Dr. Santi Wulan Purnami, S.Si., M.Si. selaku Ketua Program Studi Sarjana yang telah memberikan fasilitas, sarana, dan prasarana.
3. Dr. Muhammad Mashuri, M.T. selaku dosen wali selama masa studi yang telah banyak memberikan saran dan arahan dalam proses belajar di Departemen Statistika.
4. Dra. Wiwiek Setya Winahju, M.S. dan Dr. Dra. Kartika Fithriasari, M.Si selaku dosen pembimbing yang telah meluangkan waktu dan dengan sangat sabar memberikan bimbingan, saran, dukungan serta motivasi selama penyusunan Tugas Akhir.
5. Prof. Drs. Nur Iriawan, MIKom., Ph.D. dan Pratnya Paramitha Oktaviana, S.Si., M.Si selaku dosen penguji yang selalu sabar dalam mengomentari serta memberikan masukan dan saran dalam penyelesaian Tugas Akhir.
6. Seluruh dosen Statistika ITS yang telah memberikan ilmu dan pengetahuan yang tak ternilai harganya, serta segenap karyawan Departemen Statistika ITS.
7. Teman-teman Statistika ITS ∑26 angkatan 2015, yang selalu memberikan dukungan kepada penulis selama ini.
8. Semua teman, relasi dan berbagai pihak yang tidak bisa penulis sebutkan namanya satu per satu yang telah membantu dalam penulisan laporan ini.

Besar harapan penulis untuk mendapatkan kritik dan saran yang membangun sehingga Tugas Akhir ini dapat memberikan manfaat bagi semua pihak yang terkait.

Surabaya, Juli 2019

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

(Halaman ini sengaja dikosongkan)

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

Candi Borobudur merupakan salah satu dari sepuluh destinasi wisata yang diprioritaskan oleh Kementerian Pariwisata (Ratman, 2016). Kemegahan Candi Borobudur telah diakui oleh dunia internasional sebagai warisan budaya dan dianggap sebagai salah satu dari tujuh keajaiban dunia. Berdasarkan hal tersebut, tentunya pengelola wisata Candi Borobudur harus selalu berupaya meningkatkan kualitasnya menjadi lebih baik sehingga mendapatkan persepsi positif dari pengunjungnya. Menurut Ismayanti (2010), terbentuknya persepsi positif mengenai daerah tujuan wisata dengan berbagai atribut-atribut pariwisatanya, pada diri wisatawan menjadi salah satu kunci untuk menjamin berkembangnya suatu destinasi wisata. Persepsi positif dapat terbentuk jika objek wisata tersebut dapat memenuhi keinginan wisatawan (Pitana & Gayatri, 2005). Persepsi atau sentimen wisatawan tersebut dapat diketahui oleh pihak pengelola wisata Candi Borobudur melalui ulasan yang kerap ditulis oleh wisatawan.

Di era pesatnya teknologi seperti saat ini, banyak orang telah memanfaatkan *internet* sebagai media untuk menyampaikan ulasan atau pendapatnya. Salah satu situs yang menyediakan fasilitas *review* di situsnya adalah *TripAdvisor*, dimana situs ini merupakan situs wisata terbesar di dunia yang memungkinkan wisatawan untuk mengoptimalkan pengalaman wisatanya. Pada situs ini terdapat lebih dari 730 juta ulasan dan opini, sehingga dapat membantu wisatawan untuk memutuskan pilihan mengenai tempat tujuan, transportasi, serta akomodasi (TripAdvisor, 2017).

Menurut Borade dkk. (2017), ulasan merupakan teks yang diberikan oleh pengguna terkait sistem layanan yang diberikan. Bagi penyedia layanan, ulasan dapat bermanfaat untuk mendapatkan informasi mengenai sistem layanan yang telah memuaskan, serta perubahan yang diperlukan untuk memperbaiki sistem pelayanan. Ulasan merupakan topik yang kerap dibahas dalam analisis senti-

men, dimana analisis sentimen termasuk salah satu bentuk aplikasi dari *text mining*. Konsep dasar dari analisis sentimen ialah mengklasifikasikan polaritas teks yang terdapat dalam kalimat atau dokumen (Reyhana dkk., 2018). Mukherjee (2018) menjelaskan bahwa analisis sentimen bertujuan untuk memanfaatkan banyaknya data yang tersedia dengan melakukan analisis terhadap teks yang sebagian besar tidak terstruktur untuk memperoleh pendapat atau sentimen pengguna tentang produk dan layanan yang tersedia. Berdasarkan hal tersebut, ulasan wisatawan merupakan hal yang perlu diperhatikan oleh pihak pengelola wisata Candi Borobudur apabila ingin lebih memajukan pariwisatanya. Namun diperlukan suatu cara untuk efisiensi proses evaluasi dari data ulasan yang ada. Analisis sentimen dapat menjadi solusi untuk menangani permasalahan tersebut.

Tersedia banyak metode klasifikasi yang dapat digunakan untuk melakukan analisis sentimen, beberapa diantaranya adalah *Support Vector Machine* (SVM) dan *K-Nearest Neighbor* (K-NN). Penelitian mengenai analisis sentimen dengan metode SVM telah dilakukan oleh Kurniawan (2017), Kurniasari (2018), dan Praptiwi (2018). Hasil dari ketiga penelitian tersebut menyimpulkan bahwa SVM dengan *kernel Radial Basis Function* (RBF) memiliki akurasi yang lebih tinggi dibandingkan *kernel Linear*. Sementara itu metode K-NN untuk analisis sentimen telah digunakan dalam penelitian yang dilakukan oleh Arianto, Affandi, dan Nugroho (2017) dan Rahman (2018). Kedua penelitian tersebut menyimpulkan bahwa metode K-NN memiliki kinerja yang baik.

Dalam *text mining* terdapat suatu teknik yang dapat menggabungkan beberapa kata sebanyak *n* yang saling berdekatan secara bersamaan, yaitu *N-gram* (Koehn, 2009). Penelitian dengan menerapkan *N-gram* pada *text classification* telah dilakukan oleh Rani, Anuradha, dan Reddy (2016) dan Marafino dkk. (2014). SVM dan K-NN merupakan metode yang digunakan oleh Rani, Anuradha, dan Reddy pada penelitiannya, kemudian disimpulkan bahwa teknik *unigram* ($n = 1$) pada metode SVM dan K-NN memiliki kinerja yang lebih baik dibandingkan *bigram* ($n = 2$) dan *trigram* ($n = 3$).

Sedangkan pada penelitian yang dilakukan oleh Marafino dkk. dengan metode SVM, diperoleh hasil bahwa *bigram* memiliki kinerja yang paling baik untuk mengklasifikasikan *clinical text*.

Untuk mengklasifikasikan sentimen wisatawan Candi Borobudur, maka dalam penelitian ini dilakukan analisis sentimen pada data ulasan wisatawan Candi Borobudur dari situs *TripAdvisor* dengan metode klasifikasi. Melalui klasifikasi, dapat diperoleh jenis sentimen secara cepat dan otomatis dari data-data ulasan wisatawan yang masuk di kemudian hari. Metode klasifikasi yang digunakan dalam penelitian ini adalah SVM dan K-NN. Berdasarkan hasil penelitian terdahulu, kedua metode ini mampu menghasilkan kinerja klasifikasi yang baik. Menurut Zhang dkk. (2018), K-NN merupakan salah satu metode klasifikasi yang paling dasar dan sederhana namun memiliki kinerja klasifikasi yang baik serta sangat populer dalam bidang *data mining* dan statistik. Adapun menurut Feldman dan Sanger (2007), SVM merupakan metode yang memiliki cara kerja cepat dan efektif untuk klasifikasi pada data teks. Selain, itu dalam penelitian ini juga akan diterapkan *N-gram* untuk melihat pengaruhnya terhadap kinerja metode SVM dan K-NN. Penerapan *N-gram* diharapkan dapat meningkatkan kinerja klasifikasi.

Sebelum melakukan klasifikasi sentimen, jumlah data di setiap kelas merupakan hal yang perlu diperhatikan. Dalam klasifikasi, data dikategorikan menjadi dua jenis, yakni *balance* dan *imbalance*. Data *imbalance* merupakan kasus khusus dalam klasifikasi, dimana data di masing-masing kelas memiliki jumlah yang tidak sama atau hampir jauh berbeda. Biasanya data *imbalance* disusun oleh kelas mayoritas dan kelas minoritas. Kelas mayoritas didefinisikan sebagai kelas dengan anggota yang berjumlah banyak, begitu pula sebaliknya dengan kelas minoritas. Dalam penelitian ini, diduga terjadi kasus data *imbalance*. Hal tersebut dapat diketahui berdasarkan informasi jumlah ulasan wisatawan Candi Borobudur yang tersedia di situs *TripAdvisor*, dimana terdapat perbedaan yang jauh antara jumlah ulasan baik dan buruk. Jauhnya perbedaan tersebut menjadi perkiraan hasil tahap *labelling* sentimen positif dan negatif

dengan *lexicon* yang diperkirakan mengalami kasus *imbalance* pula. Oleh karena itu dalam penelitian ini juga akan diterapkan suatu metode untuk mengatasi kasus data *imbalance*, yakni *Synthetic Minority Oversampling Technique* (SMOTE).

Dengan menerapkan metode SVM dan K-NN yang disertai *N-gram* pada penelitian ini, akan diperoleh hasil klasifikasi sentimen wisatawan Candi Borobudur berdasarkan data ulasan yang tersedia di situs *TripAdvisor*. Hasil akhir yang diperoleh dari penelitian ini diharapkan dapat membantu pengelola wisata Candi Borobudur untuk efisiensi proses evaluasi, yakni mendapatkan jenis sentimen wisatawan secara cepat dan otomatis. Selain itu, dapat dijadikan sebagai tambahan masukan mengenai hal yang harus diperbaiki atau dipertahankan untuk memaksimalkan kepuasan wisatawan.

1.2 Rumusan Masalah

Bagi pihak pengelola wisata Candi Borobudur, ulasan atau persepsi wisatawan merupakan hal yang penting untuk diperhatikan karena dapat menjadi pertimbangan dalam menentukan kebijakan di kemudian hari. Klasifikasi data ulasan wisatawan Candi Borobudur berdasarkan jenis sentimen akan membuat proses evaluasi semakin efektif dan efisien. Oleh karena itu diperlukan suatu teknik untuk memperoleh sentimen dari wisatawan secara otomatis dan cepat berdasarkan data ulasan di situs *TripAdvisor*. Permasalahan ini dapat diselesaikan dengan menerapkan analisis sentimen.

Dalam analisis sentimen, kinerja klasifikasi dari *classifier* merupakan acuan untuk memilih metode terbaik. Tersedia banyak metode klasifikasi yang dapat digunakan, beberapa diantaranya yaitu SVM dan K-NN. Metode SVM berhasil memberikan hasil prediksi yang baik dalam masalah klasifikasi. Sedangkan K-NN merupakan metode klasifikasi yang paling dasar dan sederhana namun tetap memiliki kinerja yang baik, serta populer dalam *data mining* dan statistik. Selain itu dalam *text mining* terdapat *N-gram* sebagai teknik untuk menggabungkan beberapa kata sebanyak *n* yang saling berdekatan secara bersamaan. Dengan menerapkan *N-gram* diharapkan dapat meningkatkan kinerja klasifikasi. Berdasarkan pa-

paran tersebut, maka dalam penelitian ini akan digunakan metode SVM dan K-NN yang disertai penerapan *N-gram* untuk mengklasifikasikan sentimen wisatawan Candi Borobudur.

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai dari penelitian ini yakni sebagai berikut.

1. Mendapatkan hasil klasifikasi sentimen positif dan negatif dengan metode *K-Nearest Neighbor* (K-NN) dan *Support Vector Machine* (SVM) yang disertai penerapan *N-gram*.
2. Memperoleh perbandingan kinerja klasifikasi antara metode *K-Nearest Neighbor* (K-NN) dan *Support Vector Machine* (SVM) ketika diterapkan *N-gram* dalam proses klasifikasinya.
3. Mendapatkan kata yang sering muncul pada sentimen positif dan negatif melalui visualisasi *word cloud*.

1.4 Manfaat Penelitian

Melalui penelitian ini, dapat diperoleh metode terbaik untuk mengklasifikasikan sentimen wisatawan Candi Borobudur serta informasi mengenai hal yang kerap dibicarakan oleh wisatawan. Dari hasil tersebut, diharapkan dapat meningkatkan efisiensi proses evaluasi pariwisata dan menjadi tambahan informasi bagi pengelola wisata Candi Borobudur mengenai hal yang harus diperbaiki atau dipertahankan sehingga dapat memaksimalkan kepuasan wisatawan.

1.5 Batasan Masalah

Batasan masalah yang digunakan dalam penelitian ini adalah sebagai berikut.

1. Data yang digunakan berupa ulasan bahasa Inggris yang diperoleh dari *website TripAdvisor*.
2. Penggunaan kamus *lexicon* untuk *labelling* sentimen positif dan negatif.
3. Jenis *kernel* yang digunakan pada metode SVM adalah *Linear* dan RBF.

(Halaman ini sengaja dikosongkan)

BAB II
TINJAUAN PUSTAKA

2.1 *Text Mining*

Text mining telah mendapatkan banyak perhatian dalam beberapa tahun terakhir karena biasa diterapkan pada jejaring sosial, catatan pasien, data asuransi kesehatan, outlet berita, dll. Hung dan Zhang (2012) mendefinisikan bahwa *text mining* mengacu pada proses penggalian pola atau pengetahuan yang bermakna dan tidak biasa dari serangkaian teks yang tidak terstruktur.

Text mining sering digunakan untuk menangani permasalahan klasifikasi, *clustering*, *information extraction*, dan *information retrival* (Berry & Kogan, 2010). Pada dasarnya, proses kerja dari *text mining* banyak mengadopsi dari *data mining*. Namun *text mining* berasal dari sekumpulan bahasa alami yang tidak terstruktur, sedangkan *data mining* berasal dari *database* yang telah terstruktur (Feldman & Sanger, 2007).

2.2 Analisis Sentimen

Analisis sentimen telah menjadi lingkup penelitian yang sangat menarik dalam *Natural Language Processing* (NLP), *data mining*, *web mining*, dan *text mining*. Tujuan dari analisis sentimen adalah menganalisis opini, evaluasi, sikap, penilaian, dan emosi seseorang terhadap suatu entitas tertentu (Liu, 2012). Pendekatan untuk analisis sentimen dilakukan dengan *supervised learning*, yakni mempelajari model klasifikasi berdasarkan serangkaian data berlabel. *Labelling* data ulasan dalam penelitian ini dilakukan dengan menerapkan kamus *lexicon*, yaitu pemberian label berupa sentimen positif dan negatif ke dalam suatu ulasan berdasarkan frasa yang menyusunnya. Untuk mendapatkan label sentimen, *lexicon* bekerja berdasarkan ketentuan pada persamaan (2.1) (Mohammad, Kiritchenko, & Zhu, 2013).

$$\text{skor} = \text{jumlah kata positif} - \text{jumlah kata negatif}. \quad (2.1)$$

Skor merupakan indikator yang digunakan untuk *labelling* sentimen dari suatu ulasan. Suatu dokumen (ulasan) akan menghasilkan

label sentimen positif bila skor > 0. Apabila skor < 0, maka ulasan tersebut akan diberi label negatif. Pada penelitian ini, daftar kata yang menunjukkan sentimen positif dan negatif diperoleh dari data kamus yang disusun oleh Hu dan Liu (2004). Beberapa contoh kata yang menunjukkan sentimen positif dan negatif ditampilkan pada Tabel 2.1.

Tabel 2.1 Contoh Kata Sentimen Positif dan Negatif

Positif	Negatif
Awesome, awe, amaze, beautiful, clean, famous, good, impressive, recommend, wonder, dll	*Abusive, annoyingly, bad, break, crowd, disappoint, expensive, ignore, poorly,* dll

Dalam keilmuan statistika, tahap *labelling* untuk dua kategori kelas dapat dilakukan menggunakan regresi logistik biner. Dengan variabel prediktor berupa kata dan respon merupakan kategori kelas (0 dan 1), maka dapat diketahui label kelas dari suatu data. Namun *labelling* dalam penelitian ini tidak menerapkan regresi logistik biner karena data yang terkumpul tidak diketahui label kelasnya, sehingga dilakukan *labelling* sentimen positif dan negatif berdasarkan *lexicon*.

2.3 *Text Preprocessing*

Septiana, Ridok, dan Dewi (2013) mendefinisikan *text preprocessing* sebagai langkah awal dalam pengolahan teks yang digunakan untuk mengubah bentuk dokumen menjadi data terstruktur, yang sesuai dengan kebutuhan dan menghasilkan nilai numerik dari kata, sehingga dapat dianalisis lebih lanjut dengan *text mining*. Selain itu, adanya *preprocessing* juga bertujuan untuk meningkatkan akurasi data (Rifqi, Maharani, & Shaufiah, 2011). Tahap *preprocessing* dalam *text mining* cukup kompleks bila dibandingkan dengan *preprocessing* untuk data berupa angka atau kategori, sebab kalimat atau teks memiliki struktur yang sangat rumit. Tahap *preprocessing* yang diterapkan dalam penelitian ini meliputi *case folding, lemmatization, stopword removal,* dan *tokenizing*. Rincian mengenai penjelasan setiap tahap *text preprocessing* tersebut adalah sebagai berikut.

1. *Case folding*, merupakan proses untuk mengubah seluruh karakter teks menjadi huruf kecil serta menghilangkan tanda baca dan angka (Weiss, 2010). *Case folding* bekerja dengan cara memproses huruf abjad dari "a" hingga "z", sehingga karakter selain huruf tersebut akan dihilangkan, seperti tanda baca titik (.), koma (,), dan angka.
2. *Lemmatization*, bertujuan untuk mengubah suatu kata yang memiliki imbuhan menjadi kata dasarnya (*lemma*) namun tetap sesuai dengan kosakata bahasa Inggris, sehingga kata tersebut tidak kehilangan makna (Toman, Tesar, & Jezek, 2006). Contoh proses *lemmatization* adalah kata *amazing* berubah menjadi *amaze*. Untuk data berbahasa Inggris, dapat memilih menggunakan *lemmatization* atau *stemming*. *Stemming* merupakan proses menghilangkan imbuhan suatu kata sehingga diperoleh *root* dari kata tersebut. Namun seringkali terjadi *overstemming* dan menyebabkan kata hasil *stemming* tersebut tidak sesuai dengan kosakata bahasa Inggris. Contoh kasus *overstemming* ialah kata *amazing* berubah menjadi *amaz*. Berdasarkan pertimbangan tersebut, maka diterapkanlah *lemmatization* dalam penelitian ini. Sedangkan dalam bahasa Indonesia, *stemming* merupakan tahap untuk memperoleh kata dasar yang paling sering digunakan.
3. *Stopwords removal*, merupakan tahap menghilangkan kosakata yang tidak termasuk kata unik atau tidak memberikan makna secara signifikan pada teks. Adapun kosakata yang dimaksud antara lain kata penghubung dan kata keterangan, seperti "*the*", "*a*", "*on*", "*is*", dll. *Stopwords removal* bermanfaat untuk mengurangi jumlah *feature* yang akan digunakan. Dalam banyak kasus, penerapan *stopwords removal* juga dapat meningkatkan akurasi klasifikasi (Toman, Tesar, & Jezek, 2006).
4. *Tokenizing*, merupakan tahap pemutusan kata per kata pada kalimat. Dengan demikian urutan *string* akan terputus menjadi potongan kata yang menyusunnya (Liu, 2010).

Dari hasil *text preprocessing* yang telah dilakukan, nantinya akan diperoleh struktur data seperti Tabel 2.2. Setiap kata yang terbentuk melalui *text preprocessing* menjadi variabel prediktor (*fea-*

ture) dan terletak di satu baris yang sama. Tambahan kata dari ulasan yang baru akan terletak pada kolom berikutnya namun masih di baris yang sama. Apabila terdapat kesamaan kata antara suatu ulasan baru dengan kata yang telah ada pada struktur data, maka kata tersebut tidak lagi ditambahkan di kolom baru. Dengan demikian tidak terdapat kata atau *feature* yang sama dalam satu struktur data.

Tabel 2.2 Ilustrasi Struktur Data setelah *Text Preprocessing* (Sumber: Tripathy, Agrawal, & Rath, 2016)

Data	**Kata (*Feature*)**				
	1	**2**	**3**	**4**	**5**
1	1	1	1	0	0
2	1	1	0	1	0
3	1	1	0	0	1

Pada ilustrasi struktur data di Tabel 2.2, baris menunjukkan ulasan, kolom menyatakan kata atau *feature*, sedangkan elemen (*i, j*) merupakan frekuensi munculnya kata ke-*j* (kolom) pada ulasan ke-*i* (baris).

2.4 *N-gram*

N-gram merupakan suatu teknik penggabungan beberapa kata sebanyak *n* yang saling berdekatan secara bersamaan (Koehn, 2009). Dalam kasus analisis sentimen, *N-gram* menganalisis sentimen suatu teks atau dokumen berdasarkan ukuran *n* yang digunakan (Tripathy, Agrawal, & Rath, 2016). *N-gram* mampu menangkap lebih banyak informasi dengan hanya meningkatkan ukuran *n*. Marafino dkk. (2014) berhipotesis bahwa penyertaan *N-gram* dapat meningkatkan kinerja klasifikasi.

N-gram memiliki ukuran yang berbeda-beda, bergantung dari ukuran *n* yang ditentukan. Apabila *n* = 1 maka disebut *unigram*, *n* = 2 disebut *bigram*, dan *n* = 3 disebut *trigram*. Sedangkan untuk ukuran *N-gram* yang lebih tinggi, biasanya disebut *four-gram*, *five-gram*, dan seterusnya. Penerapan *N-gram* pada contoh kalimat "*very nice and helpful staff*" adalah sebagai berikut.

1. *Unigram*: "*very*", "*nice*", "*and*", "*helpful*", "*staff*".
2. *Bigram*: "*very nice*", "*nice and*", "*and helpful*", "*helpful staff*".

3. *Trigram*: "*very nice and*", "*nice and helpful*", "*and helpful staff*".

2.5 *Term Weighting*

Pembobotan kata atau *term weighting* merupakan tahapan penting dalam *text mining*. T*erm weighting* berguna untuk mengukur nilai suatu kata dalam keseluruhan data ulasan (Khair, 2009). Tahap *term weighting* dilakukan terhadap kata-kata yang telah melalui *text preprocessing*. Salah satu metode *term weighting* yang sering digunakan adalah *Term Frequency-Inverse Document Frequency* (TF-IDF).

TF-IDF merupakan metode pembobotan yang tergolong ke dalam prosedur statistik. Meskipun TF-IDF dapat dikatakan sebagai metode lama, namun metode ini memiliki cara kerja yang sederhana dan efektif (Salton & Buckley, 1988). Melalui TF-IDF, dapat diketahui tingkat kepentingan suatu kata terhadap kumpulan *review* (Garreta & Moncecchi, 2013).

TF-IDF bekerja dengan cara melibatkan perkalian antara *Term Frequency* (TF) dengan *Inverse Document Frequency* (IDF). TF menunjukkan jumlah kemunculan sebuah kata pada suatu *review*. Sedangkan IDF mengukur kemunculan sebuah kata pada seluruh kumpulan ulasan (Tripathy, Agrawal, & Rath, 2016). Formula yang digunakan untuk mendapatkan nilai TF-IDF dituliskan ke dalam persamaan (2.2).

$$w_{ij} = tf_{ij} \times idf \text{ , dengan } idf = \log\left(\frac{N}{df_i}\right), \tag{2.2}$$

keterangan:

w_{ij} = bobot dari kata *j* pada ulasan ke-*i*

N = jumlah seluruh ulasan

tf_{ij} = jumlah munculnya kata *j* pada ulasan *i*

df_i = banyaknya ulasan yang mengandung kata *j*

i = 1, 2, ..., M

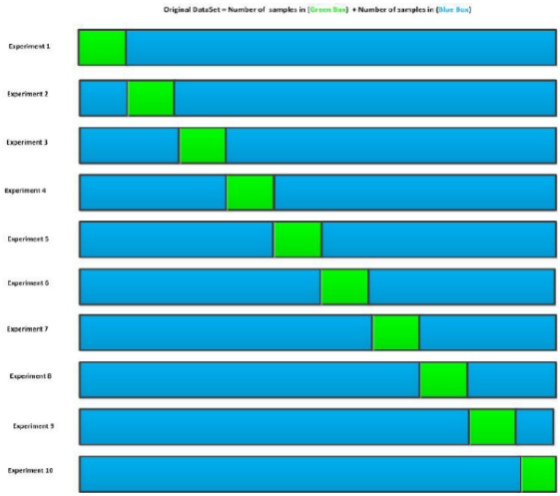
j = 1, 2, ..., m.

Manning, Raghavan, dan Schutze (2008) menjelaskan bahwa bobot TF-IDF akan bernilai tinggi ketika suatu kata muncul berulang kali dalam sejumlah kecil dokumen, sehingga memberikan bobot yang tinggi pada dokumen tersebut. Sedangkan kata yang muncul di hampir seluruh dokumen akan memiliki nilai TF-IDF yang rendah.

Setelah diperoleh nilai bobot dari setiap kata, maka variabel prediktor (X) atau *feature* yang digunakan tidak lagi berupa frekuensi kemunculan kata, melainkan nilai bobot TF-IDF.

2.6 *K-fold Cross Validation* (*K-fold* CV)

K-fold CV merupakan salah satu metode yang berfungsi untuk mempartisi data menjadi *training* dan *testing*. Banyak peneliti yang menerapkan metode ini karena dapat mengurangi bias yang terjadi dalam pengambilan sampel. *K-fold* CV bekerja dengan cara membagi data *training* dan *testing* secara berulang sebanyak *K* kali, dimana setiap data memiliki kesempatan untuk menjadi data *testing*. *K* didefinisikan sebagai angka partisi data yang digunakan untuk pembagian *training-testing* (Gokgoz & Subasi, 2015).



Gambar 2.1 Ilustrasi 10-*fold* CV
(Sumber: Babatunde dkk., 2015)

Penentuan nilai *K* merupakan hal yang penting dalam *K-fold* CV. Nilai *K* yang biasa digunakan adalah 5 atau 10, tetapi tidak terdapat aturan pasti mengenai hal tersebut (Kuhn & Johnson, 2013). Sedangkan dalam *machine learning*, nilai *K* yang sangat umum digunakan adalah 10 karena menghasilkan bias yang rendah. Ilustrasi *K-fold* CV dengan nilai *K* = 10 ditunjukkan pada Gambar 2.1.

2.7 *Synthetic Minority Oversampling Technique* (SMOTE)

Umumnya, klasifikasi akan memiliki kinerja buruk ketika menghadapi kasus *dataset* yang tidak seimbang (*imbalance*). Bila dalam klasifikasi tetap memaksa untuk menggunakan data *imbalance*, maka akan dihasilkan kelas yang salah terklasifikasi (misklasifikasi).

Salah satu metode untuk menangani kasus data *imbalance* adalah *Synthetic Minority Oversampling Technique* (SMOTE), dimana metode ini diusulkan oleh Chawla dkk. SMOTE merupakan suatu metode *oversampling* yang bekerja dengan menerapkan *sampling based approach*. Pendekatan *sampling* melakukan modifikasi terhadap data *training* sehingga kedua kelas dapat dipresentasikan dengan baik. Adapun metode *oversampling* dilakukan untuk menyeimbangkan jumlah data dengan cara meningkatkan jumlah data di kelas minor. Penambahan jumlah data di kelas minoritas diperoleh melalui replikasi data secara acak, sehingga nantinya jumlah data di kelas minoritas akan sama dengan kelas mayoritas.

Chawla dkk. (2002) menjabarkan bahwa cara kerja SMOTE adalah membuat data *synthetic* berdasarkan tetangga terdekat, dimana pemilihan tetangga terdekat ini didasarkan pada jarak *euclidean* antara dua data. Untuk kasus *text mining*, data *synthetic* merupakan nilai bobot baru dari suatu kata yang terbentuk melalui replikasi data di kelas minor. Pembuatan data *synthetic* dapat dirumuskan dengan persamaan (2.3) (Sain & Purnami, 2015).

$$x_{syn} = x_i + (x_{knn} - x_i) \times \delta , \tag{2.3}$$

keterangan:

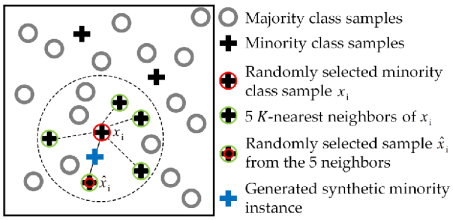
x_{syn} = nilai bobot *synthetic*

x_i = nilai bobot data ke-*i* di kelas minor

δ = bilangan acak antara 0 dan 1

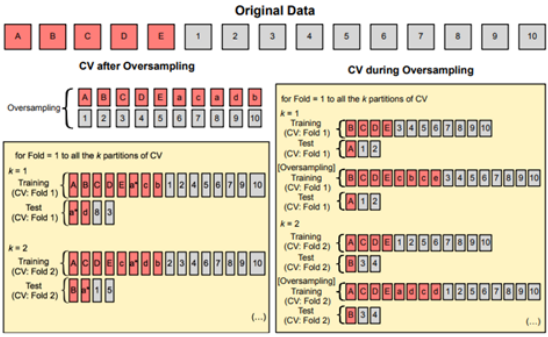
x_{knn} = nilai bobot dari data di kelas minor yang memiliki jarak terdekat dengan x_i.

Gambar 2.2 menunjukkan ilustrasi pembangkitan data *synthetic* dengan metode SMOTE.



Gambar 2.2 Konsep Dasar Metode SMOTE
(Sumber: Ma dkk., 2019)

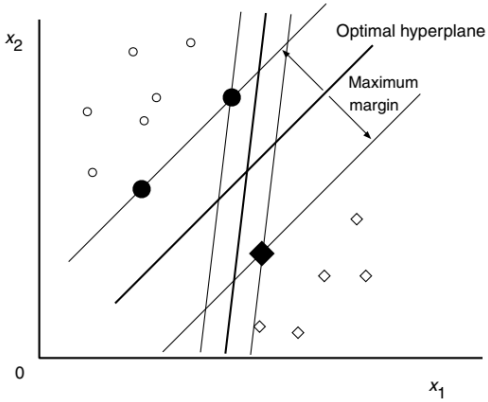
Pada prosedur *K-fold* CV, SMOTE dilakukan di masing-masing data *training* pada setiap *fold.* Apabila SMOTE dilakukan sebelum prosedur *K-fold* CV, maka akan diperoleh hasil yang *over-optimistic*, yaitu hasil klasifikasi menjadi terlalu baik dikarenakan terdapat kemungkinan bahwa data replikasi berada pada data *testing* (Santos dkk., 2018). Prosedur *K-fold* CV dengan penerapan SMOTE diilustrasikan pada Gambar 2.3.



Gambar 2.3 Ilustrasi *K-fold* CV dengan SMOTE
(Sumber: Santos dkk., 2018)

2.8 *Support Vector Machine* (SVM)

SVM adalah salah satu metode klasifikasi biner berdasarkan strategi *margin* maksimum yang dikenalkan oleh Vapnik. Metode ini berhasil memberikan prediksi yang baik untuk kasus klasifikasi dan juga regresi. Secara sederhana, konsep SVM dapat dijelaskan sebagai pencarian *hyperplane* terbaik yang memiliki sebuah fungsi untuk memisahkan dua kelas pada *input space*. Penentuan *hyperplane* optimal pada SVM ditampilkan pada Gambar 2.4.

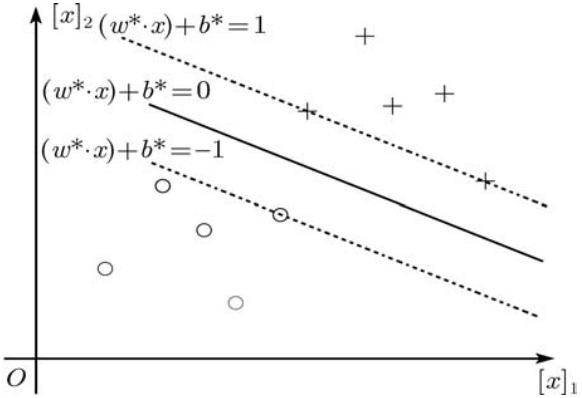


Gambar 2.4 *Hyperplane* Optimal dengan SVM
(Sumber: Abe, 2010)

Hyperplane dikatakan optimum apabila telah menghasilkan *margin* terbesar. *Margin* didefinisikan sebagai jarak antara *hyperplane* terhadap data *input* yang dekat dengan *hyperplane* (*support vector*) (Abe, 2010). Prinsip dasar dari SVM adalah klasifikasi linier, kemudian terdapat pengembangan sehingga dapat mengatasi kasus non-linier, yakni dengan melibatkan *kernel trick*.

2.8.1 SVM pada *Linearly Separable Data*

Dimisalkan terdapat *m* dimensi data *input* $\mathbf{x}_i$ (i = 1, 2, ..., M), yakni vektor pembobotan TF-IDF dari kata pada ulasan ke-*i*, dan $y_i \in \{+1, -1\}$ adalah kelas dari ulasan ke-*i*. Apabila data tersebut terpisah secara linier, maka ilustrasi klasifikasi SVM dua dimensi untuk sepasang variabel X ditampilkan pada Gambar 2.5.



Gambar 2.5 Ilustrasi SVM pada *Linearly Separable Data*
(Sumber: Deng, Tian, & Zhang, 2012)

Ilustrasi pada Gambar 2.5 menunjukkan bahwa terdapat dua kelas yang dipisahkan oleh sepasang bidang pembatas yang sejajar. Bidang pembatas pertama membatasi kelas (1), sedangkan kelas (-1) dibatasi oleh bidang kedua. Fungsi pemisah untuk kedua kelas tersebut dituliskan ke dalam bentuk pertidaksamaan (2.4).

$$\mathbf{w}'\mathbf{x}_i + b \begin{cases} \geq 1; y_i = 1 \\ \leq -1; y_i = -1. \end{cases} \tag{2.4}$$

Notasi **w** merupakan vektor bobot, sedangkan *b* adalah bias.

Jarak antara masing-masing bidang pembatas terhadap *hyperplane* adalah $\frac{1}{\|\mathbf{w}\|}$, dengan demikian diperoleh *margin* sebesar $\frac{2}{\|\mathbf{w}\|}$ (Webb & Copsey, 2011). Notasi $\|\mathbf{w}\|$, yang tidak lain adalah $\sqrt{\mathbf{w}'\mathbf{w}} = \sqrt{w_1^2 + w_2^2 + ... + w_m^2}$, merupakan jarak *Euclidean* (*norm Euclidean*) dari **w**. *Hyperplane* yang optimal dapat diperoleh dengan dua cara, yaitu memaksimumkan fungsi objektif $\frac{2}{\|\mathbf{w}\|}$ atau

meminimumkan fungsi objektif $\frac{1}{2}\|\mathbf{w}\|^2$. Umumnya *hyperplane* yang optimal didapatkan dengan meminimumkan fungsi objektif $\frac{1}{2}\|\mathbf{w}\|^2$ terhadap *constraint* $y_i(\mathbf{w}'\mathbf{x}_i + b) - 1 \geq 0$. Pendekatan standar yang digunakan untuk menyelesaikan optimasi tersebut adalah formula *Lagrange primal* (L_P) yang diformulasikan pada persamaan (2.5) (Fletcher, 1987).

$$L_P(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{M} \alpha_i \{y_i(\mathbf{w}'\mathbf{x}_i + b) - 1\}, \tag{2.5}$$

dengan $\alpha_i \geq 0$ merupakan nilai koefisien *Lagrange*. Penyelesaian optimal dari persamaan (2.5) dilakukan dengan cara meminimumkan L_P terhadap $\mathbf{w}$ dan b. Dengan demikian diperoleh persamaan (2.6) dan (2.7).

$$\mathbf{w} = \sum_{i=1}^{M} \alpha_i y_i \mathbf{x}_i \,, \tag{2.6}$$

$$\sum_{i=1}^{M} \alpha_i y_i = 0 \,. \tag{2.7}$$

Apabila persamaan (2.6) dan (2.7) disubstitusikan ke dalam persamaan (2.5), maka akan dihasilkan formula *Lagrange dual* (L_D) seperti yang diformulasikan pada persamaan (2.8).

$$\begin{aligned} L_D &= \sum_{i=1}^{M} \alpha_i - \frac{1}{2}\left(\sum_{i,h=1}^{M} \alpha_i \alpha_h y_i y_h \mathbf{x}_i{}' \mathbf{x}_h \right) \\ &= \sum_{i=1}^{M} \alpha_i - \frac{1}{2}\left(\sum_{i=1}^{M} \alpha_i y_i \mathbf{x}_i \right)' \left(\sum_{i=1}^{M} \alpha_i y_i \mathbf{x}_i \right). \end{aligned} \tag{2.8}$$

Kemudian persamaan (2.8) digunakan untuk mendapatkan nilai α_i, yang diperoleh dengan cara memaksimumkan L_D terhadap α, de-

ngan syarat $\sum_{i=1}^{M} \alpha_i y_i = 0$ dan $\alpha_i \geq 0$. Berdasarkan penjabaran tersebut, permasalahan pencarian *hyperplane* terbaik dapat dirumuskan menjadi persamaan (2.9).

$$\max_{\alpha} L_D = \max\left(\sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,h=1}^{M} \alpha_i \alpha_h y_i y_h \mathbf{x}_i' \mathbf{x}_h \right). \quad (2.9)$$

Setiap data *input* dengan $\alpha_i > 0$ dianggap sebagai *support vector*, sedangkan data *input* dengan nilai $\alpha_i = 0$ disebut *non-support vector*. Selanjutnya hasil nilai α_i digunakan untuk mendapatkan **w**. Dengan demikian diperoleh persamaan (2.10) yang merupakan *rules* klasifikasi metode SVM untuk *linearly separable data*.

$$f(\mathbf{x}) = \sum_{i=1}^{sv} \alpha_i y_i \mathbf{x}_i' \mathbf{x} + b, \quad (2.10)$$

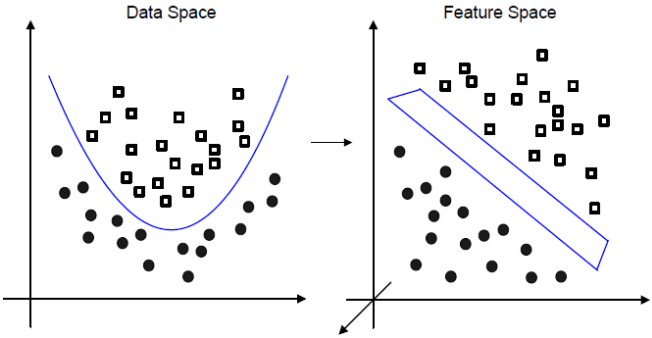
dengan i = 1, 2, ..., sv (banyaknya *support vector*), $\mathbf{x}_i$ merupakan *support vector*, dan $b = \frac{1}{2}[(\mathbf{w}.\mathbf{x}^*(1)) + (\mathbf{w}.\mathbf{x}^*(-1))]$. Notasi $\mathbf{x}^*(1)$ menunjukkan *support vector* untuk kelas +1 dan $\mathbf{x}^*(-1)$ untuk kelas -1 (Vapnik & Chervonenkis, 1974).

2.8.2 SVM pada *Non-Linearly Separable Data*

Hardle, Prastyo, dan Hafner (2014) menjelaskan bahwa pada kasus riil, tidak semua data dapat dipisahkan secara linier. Hal tersebut menyebabkan kesulitan dalam menemukan bidang pemisah secara linier. Oleh karena itu diperlukan formula SVM yang dimodifikasi untuk menemukan solusi dari permasalahan tersebut. SVM untuk kasus *non-linearly separable data* ditunjukkan melalui ilustrasi pada Gambar 2.6.

Ilustrasi pada Gambar 2.6 bagian kiri, terlihat bahwa kelas lingkaran dan persegi tidak dapat dipisahkan secara linier apabila menggunakan ruang berdimensi dua. Namun pada Gambar 2.6 bagian kanan, ruang berdimensi lebih tinggi dapat memisahkan kelas lingkaran dan persegi secara linier melalui *hyperplane*. Berdasar-

kan hal tersebut maka dilakukan modifikasi dengan menambahkan fungsi *kernel* ketika dilakukan pencarian *hyperplane* yang optimal.



Gambar 2.6 Ilustrasi SVM pada *Non-Linearly Separable Data*
Sumber: (Hardle, Prastyo, & Hafner, 2014)

Fungsi *kernel* pada persamaan (2.11) merupakan *scalar product* antar vektor, yang bekerja dengan cara mentransformasi data *input* ke dalam *feature space* berdimensi lebih tinggi sehingga dapat dipisahkan secara linier.

$$K(\mathbf{x}_i, \mathbf{x}_h) = \phi(\mathbf{x}_i)'\phi(\mathbf{x}_h). \tag{2.11}$$

Adanya tambahan fungsi *kernel* mengakibatkan persoalan optimasi di persamaan (2.9) berubah menjadi persamaan (2.12).

$$\max_{\alpha} L_D = \max\left(\sum_{i=1}^{M}\alpha_i - \frac{1}{2}\sum_{i,h=1}^{M}\alpha_i\alpha_h y_i y_h K(\mathbf{x}_i, \mathbf{x}_h)\right). \tag{2.12}$$

Selanjutnya persamaan (2.12) digunakan untuk memaksimalkan α_i, dengan syarat $\sum_{i=1}^{M}\alpha_i y_i = 0$ dan $0 \le \alpha_i \le C$. Dengan demikian diperoleh persamaan (2.13) yang digunakan untuk klasifikasi pada kasus *non-linearly separable* (Webb & Copsey, 2011).

$$f(\mathbf{x}) = \sum_{i=1}^{sv}\alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b. \tag{2.13}$$

Notasi α_i adalah nilai koefisien *Lagrange*, $\mathbf{x}_i$ merupakan *support vector*, y_i menunjukkan kelas *support vector* (+1 atau -1), $K(\mathbf{x}, \mathbf{x}_i)$ merupakan fungsi *kernel*, dan *b* adalah bias. Kemudian indeks *i* = 1, 2, ..., *sv* (banyaknya *support vector*).

Lima jenis *kernel* yang umum digunakan pada metode SVM ditampilkan pada Tabel 2.3 (Kecman, 2005).

Tabel 2.3 *Kernel* Metode SVM

Kernel	Fungsi Kernel
Linear	$K(\mathbf{x}, \mathbf{x}_i) = \mathbf{x}'\mathbf{x}_i$
Polynomial	$K(\mathbf{x}, \mathbf{x}_i) = [(\mathbf{x}'\mathbf{x}_i) + 1]^d$
Radial Basis Function (RBF) atau *Gaussian*	$K(\mathbf{x}, \mathbf{x}_i) = \exp(-\gamma \parallel \mathbf{x} - \mathbf{x}_i \parallel^2), \gamma > 0$
Sigmoid	$K(\mathbf{x}, \mathbf{x}_i) = \tanh[(\mathbf{x}'\mathbf{x}_i) + b]$
Inverse Multiquadric Function	$K(\mathbf{x}, \mathbf{x}_i) = \frac{1}{\sqrt{\parallel \mathbf{x} - \mathbf{x}_i \parallel^2 + \beta}}$

Notasi $\mathbf{x}_i$ pada Tabel 2.3 merupakan *support vector* dan *gamma* (γ) menunjukkan parameter *kernel* RBF.

2.9 *K-Nearest Neighbor* (K-NN)

K-NN merupakan salah satu metode klasifikasi yang paling dasar dan sederhana (Zheng dkk., 2015). Meskipun demikian, metode K-NN tetap memiliki kinerja klasifikasi yang baik, serta sangat populer dalam *data mining* dan statistik (Zhang dkk., 2018). K-NN tidak membutuhkan pembentukan model dikarenakan seluruh data *training* digunakan untuk memprediksi data *testing*. K-NN memiliki konsep algoritma yang sederhana. Untuk mengklasifikasikan data *testing*, dilakukan pencarian *k* tetangga terdekat antara data *testing* dan seluruh data *training*, kemudian menggunakan kategori dari *k* tetangga terdekat tersebut untuk menentukan kategori data *testing* (Sun, Wong, & Kamel, 2009).

Kinerja dari metode K-NN sangat dipengaruhi oleh nilai *k*. Jika *k* memiliki nilai yang terlalu besar, maka batasan antara setiap klasifikasi menjadi semakin kabur, namun dapat mengurangi efek

noise. Dalam praktiknya, nilai *k* biasanya dioptimalkan melalui banyak uji coba pada data *testing*. Penentuan nilai *k* dapat dimulai dari *k* = 1, lalu nilai *k* ditingkatkan hingga diperoleh *error* yang paling minimum (Han, Kamber, & Pei, 2011). Selain itu, dapat pula ditinjau berdasarkan nilai akurasi dari setiap nilai *k* yang diujicobakan (Ma, Yang, & Cheng, 2014).

Di bawah ini merupakan rincian langkah-langkah klasifikasi metode K-NN.

1. Menentukan nilai *k*.
   Nilai *k* diuji coba mulai dari *k* = 1 hingga senilai *k* tertentu, lalu dipilih nilai *k* yang memiliki kinerja klasifikasi paling optimal.
2. Menghitung jarak antara data *training* dengan *testing*.
   Salah satu rumus jarak yang umum digunakan adalah *euclidean*, yang dirumuskan dengan persamaan (2.14).

$$d(1,2) = \sqrt{\sum_{j=1}^{m} (x_{1j} - x_{2j})^2}\ , \qquad (2.14)$$

   dengan $d(1,2)$ menunjukkan jarak antara data *testing* dengan *training*, x_{1j} adalah nilai bobot TF-IDF pada data *training*, x_{2j} adalah nilai bobot TF-IDF pada data *testing*, dan *m* me-rupakan jumlah *feature* (kata).
3. Mengurutkan jarak terdekat.
   Hasil dari seluruh perhitungan jarak *euclidean* diurutkan berdasarkan nilai yang terendah, sehingga nantinya diperoleh tetangga terdekat antara data *testing* dengan *training*.
4. Klasifikasi data *testing*.
   Data *testing* akan diklasifikasikan ke dalam kelas yang memiliki anggota terbanyak pada *k* tetangga terdekat.

2.10 Evaluasi Kinerja Klasifikasi

Hasil dari suatu *classifier* dievaluasi untuk mengukur kemampuan *classifier* tersebut dalam mengklasifikasikan dokumen ke dalam kelas yang tepat. Beberapa perhitungan yang sering digunakan untuk mengukur kinerja klasifikasi adalah akurasi, sensitivitas (*recall*), dan spesifisitas (Bekkar, Djemaa, & Alitouche, 2013).

Untuk kelas biner, ketiga ukuran evaluasi tersebut dirumuskan berdasarkan tabel *confusion matrix* seperti yang ditampilkan pada Tabel 2.4.

Tabel 2.4 *Confusion Matrix*

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

Rumus untuk mengevaluasi hasil klasifikasi diformulasikan pada persamaan (2.15), (2.16), dan (2.17).

$$\text{akurasi} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}}, \tag{2.15}$$

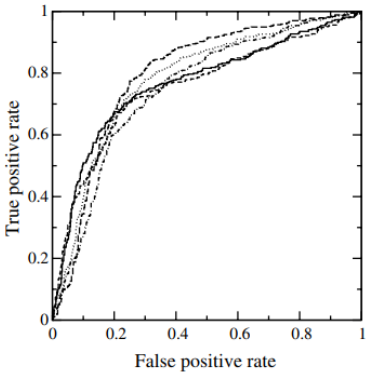
$$\text{sensitivitas} = \frac{\text{TP}}{\text{TP}+\text{FN}}, \tag{2.16}$$

$$\text{spesifisitas} = \frac{\text{TN}}{\text{TN}+\text{FP}}. \tag{2.17}$$

Akurasi merupakan nilai yang mengukur efektivitas secara keseluruhan dari suatu *classifier*. Sensitivitas (*recall*) menilai kemampuan suatu *classifier* untuk mengklasifikan dokumen ke kelas positif dengan tepat. Adapun spesifisitas merupakan ukuran untuk menilai efektifitas suatu *classifier* dalam hal mengidentifikasi label negatif (Sokolova & Lapalme, 2009). Namun akurasi tidak lagi dapat dijadikan sebagai tolok ukur evaluasi ketika ditemui kasus data *imbalance*, karena kelas minoritas tidak memiliki dampak yang berarti pada akurasi (Sun, Wong, & Kamel, 2009). Oleh sebab itu digunakan formula lain yang dapat diterapkan pada data *imbalance*, yakni dengan menggabungkan unsur sensitivitas dan spesifisitas. Salah satu ukuran kinerja klasifikasi yang menggabungkan dua unsur tersebut adalah *Area Under the Curve* (AUC). Nilai AUC dapat diperoleh melalui persamaan (2.18).

$$\text{AUC} = \frac{1}{2}\left(\frac{\text{TP}}{\text{TP}+\text{FN}}+\frac{\text{TN}}{\text{TN}+\text{FP}}\right). \tag{2.18}$$

AUC merupakan indikator performansi kurva *Receiver Operating Characteristic* (ROC) yang meringkas kinerja suatu *classifier* menjadi satu nilai. Kurva ROC diilustrasikan pada Gambar 2.7.



Gambar 2.7 Ilustrasi Kurva ROC
(Sumber: Fawcett, 2006)

AUC lebih dipertimbangkan dalam penilaian model karena kinerja suatu metode klasifikasi diurutkan berdasarkan perolehan nilai AUC. Umumnya nilai AUC berkisar antara 0,5 – 1,0, dengan interpretasi setiap rentang nilai AUC ditampilkan pada Tabel 2.5 (Bekkar, Djemaa, & Alitouche, 2013).

Tabel 2.5 Kriteria Nilai AUC

Nilai AUC	Keterangan
0,5 – 0,6	*Poor*
0,6 – 0,7	*Fair*
0,7 – 0,8	*Good*
0,8 – 0,9	*Very Good*
0,9 – 1,0	*Excellent*

2.11 *Word Cloud*

Word cloud merupakan teknik yang sering digunakan untuk visualisasi data teks. Melalui *word cloud*, dapat diketahui kata-kata yang sering muncul dalam dokumen berdasarkan ukuran hurufnya. Semakin besar ukuran kata, maka semakin sering pula kata tersebut muncul dalam dokumen (Castella & Sutton, 2014). Ilustrasi *word*

cloud untuk *unigram*, *bigram*, dan *trigram* (subbab 2.4) ditunjukkan pada Gambar 2.8.

a) **b)** **c)**

Gambar 2.8 *Unigram*, *Bigram* dan *Trigram Word Cloud*
(Sumber: Castella & Sutton, 2014; Syaani, 2019; Shaheen, Ahsan, & Anwar, 2018)

2.12 *TripAdvisor*

TripAdvisor merupakan situs wisata terbesar di dunia yang memungkinkan wisatawan untuk merencanakan dan memperoleh pengalaman berwisata yang optimal. Terdapat lebih dari 730 juta ulasan dan opini yang terdapat di situs ini sehingga dapat membantu wisatawan untuk mengambil keputusan dalam memilih tempat tujuan, transportasi, serta akomodasi. Situs *TripAdvisor* telah tersedia dalam 49 *domain* dan rata-rata pengunjung situs ini mencapai 490 juta pengunjung setiap bulan (TripAdvisor, 2017). Melalui situs ini, seseorang yang telah atau sedang melakukan *travelling* dapat berbagi pengalamannya dengan cara memberi ulasan dan mem-

posting foto mengenai tempat yang dikunjungi (Chung & Buhalis, 2008).

2.13 Candi Borobudur

Candi Borobudur merupakan candi bercorak Buddha terbesar di dunia yang berada di Kabupaten Magelang, Provinsi Jawa Tengah. Dibangun pada abad ke-9, candi ini memiliki total area sebesar 2500 m^2. Candi Borobudur merupakan salah satu monumen kuno yang paling terpelihara di Indoesia dan paling sering dikunjungi oleh lebih dari satu juta wisatawan, baik mancanegara maupun domestik. Kemegahan Candi Borobudur telah diakui oleh dunia internasional sebagai jenis warisan budaya dan dianggap sebagai salah satu dari tujuh keajaiban dunia (Kementerian Pariwisata, 2018). Pada tahun 1970-an, Pemerintah Indonesia, dengan bantuan dana dari *United Nations Educational, Scientific and Cultural Organization* (UNESCO), melakukan pemugaran terhadap Candi Borobudur. Dibutuhkan waktu sekitar delapan tahun untuk menyelesaikan pemugaran tersebut, hingga akhirnya Candi Borobudur resmi dibuka oleh presiden Indonesia pada tanggal 23 Februari 1983. Sejak tahun 2016, Candi Borobudur ditetapkan sebagai satu dari sepuluh destinasi wisata yang diprioritaskan oleh Kementerian Pariwisata (Ratman, 2016).

2.14 *Web Scraping*

Web scraping merupakan teknik pengambilan sebuah dokumen dari suatu *website* yang dibangun dengan bahasa *markup*, seperti *Hyper Text Markup Language* (HTML). *Web scraping* bermanfaat untuk mendapatkan informasi dari suatu *website*, baik secara keseluruhan atau sebagian, sehingga dapat digunakan untuk kepentingan tertentu (Johnson & Gupta, 2012). Secara umum terdapat empat tahap yang perlu dilakukan dalam *web scraping*, yakni sebagai berikut (Turland, 2010).

1. *Create scraping template*, yaitu mengamati dokumen HTML dari *website* yang akan diambil informasinya. *Tag* HTML juga dilakukan dalam tahap ini untuk mengapit informasi yang diperlukan.

2. *Explore site navigation*, merupakan proses menelusuri mekanisme navigasi pada *website* tersebut untuk ditirukan pada program *web scraper* yang akan dibuat.
3. *Automate navigation and extraction*, yakni otomatisasi pengambilan informasi dari *website* berdasarkan hasil tahap 1 dan 2.
4. *Extracted data and package history*, yakni menyimpan data hasil *scraping* pada tahap 3 ke dalam suatu *database*.

BAB III
METODOLOGI PENELITIAN

3.1 Sumber Data

Penelitian ini menggunakan data sekunder yang didapatkan melalui teknik *scraping* pada *website TripAdvisor*. Data yang dikumpulkan berupa ulasan atau *review* bahasa Inggris mengenai wisata Candi Borobudur, sejak Oktober 2005 - 11 Maret 2019. Beberapa contoh data ulasan dalam penelitian ini ditampilkan pada Tabel 3.1.

Tabel 3.1 Contoh Data Ulasan

No.	Ulasan
1.	*A UNESCO world heritage site, this place is well preserved and so clean. Excellent toilet facilities. A combine ticket for Borobudur and Prambanan temples (valid for 2 consecutive days) makes a cheaper option ...*
2.	*It is a very large Buddhist temple, you need to be early around 8:00 when there is not so much tourists. The visit took me around hour and half without guide. If you have guide it can be a lot more ...*
⋮	⋮
3591	*The history and the architecture at this site is absolutely amazing! I'll always remember the day we spent there as an incredible journey back in time.*

3.2 Variabel Penelitian dan Struktur Data

Terdapat dua variabel yang digunakan pada penelitian ini, yakni prediktor (X) atau *feature* dan respon (Y). Variabel prediktor merupakan frekuensi kemunculan kata, sedangkan variabel respon adalah jenis sentimen (positif dan negatif). Tabel 3.2 menunjukkan rincian kedua variabel tersebut.

Tabel 3.2 Variabel Penelitian

Variabel	Keterangan	Skala Data
Y	Kelas sentimen 0 = Positif 1 = Negatif	Nominal
X	Kata (Frekuensi)	Rasio

Struktur data setelah dilakukan *text preprocessing* dan *labelling* ditunjukkan pada Tabel 3.3.

Tabel 3.3 Struktur Data

No. Ulasan	Kelas Sentimen (y)	Kata 1	⋯	Kata m
1	y_1	$x_{1,1}$	⋯	$x_{1,m}$
2	y_2	$x_{2,1}$	⋯	$x_{2,m}$
⋮	⋮	⋮	⋮	⋮
M	y_M	$x_{M,1}$	⋯	$x_{M,m}$

3.3 Langkah Analisis

Langkah analisis yang diterapkan pada penelitian ini adalah sebagai berikut.

1. Mengumpulkan data ulasan berbahasa Inggris wisatawan Candi Borobudur di situs *TripAdvisor* menggunakan teknik *web scraping* berdasarkan tahap-tahap di subbab 2.14.
2. Melakukan *text preprocessing* untuk mengubah data teks menjadi lebih terstruktur dan menghasilkan nilai numerik, dengan tahap-tahap sebagai berikut.

a. *Case folding*, yaitu mengubah seluruh teks menjadi huruf kecil (non kapital), serta menghilangkan angka dan tanda baca.
b. *Lemmatization*, yaitu mendapatkan kata dasar (*lemma*) di suatu kata namun tetap sesuai dengan kosakata bahasa Inggris.
c. Menghapus kata yang teridentifikasi dalam *stopwords list*. Penelitian ini menggunakan *stopwords list* yang dipublikasikan oleh *University of Glasgow*.
d. *Tokenizing*, yaitu pemecahan kalimat ulasan menjadi kata per kata.

3. *Labelling* sentimen positif dan negatif terhadap data ulasan berdasarkan kamus *lexicon* yang disusun oleh Hu & Liu (2004).
4. Membentuk *N-gram* dengan ukuran $n = 1$ (*unigram*), $n = 2$ (*bigram*), dan $n = 3$ (*trigram*).
5. Melakukan analisis karakteristik data ulasan wisatawan. Karakteristik yang dianalisis adalah lima kata dengan frekuensi tertinggi di *unigram*, *bigram*, dan *trigram*. Selain itu hasil *labelling* sentimen berdasarkan *lexicon* juga divisualisasikan mengguna-

kan *pie chart*. Hal tersebut dilakukan untuk mengetahui persentase sentimen positif dan negatif pada data ulasan wisatawan.

6. Melakukan pembobotan TF-IDF untuk masing-masing *N-gram*. Nilai bobot dari setiap *feature* diperoleh dengan menerapkan persamaan (2.2).
   Setelah diperoleh nilai bobot dari setiap kata, maka variabel prediktor (X) atau *feature* yang digunakan tidak lagi berupa frekuensi kemunculan kata, melainkan nilai hasil pembobotan TF-IDF, yang dinotasikan dengan w_{ij}. Dengan demikian didapatkan struktur data seperti pada Tabel 3.10.

Tabel 3.10 Struktur Data setelah *Term Weighting*

No. Ulasan	Kelas Sentimen (y)	Kata 1	⋯	Kata m
1	y_1	$w_{1,1}$	⋯	$w_{1,m}$
2	y_2	$w_{2,1}$	⋯	$w_{2,m}$
⋮	⋮	⋮	⋮	⋮
M	y_M	$w_{M,1}$	⋯	$w_{M,m}$

7. Mempartisi data menjadi *training-testing* dengan metode *K-fold Cross Validation* di setiap *N-gram*.
   Nilai K yang digunakan adalah K = 10, dengan perbandingan antara data *training-testing* sebesar 90%:10%.
8. Membuat data *synthetic* di data *training* menggunakan metode SMOTE pada kelas minoritas untuk setiap *N-gram*, dengan menerapkan rumus di persamaan (2.3).
9. Klasifikasi sentimen wisatawan Candi Borobudur menggunakan metode K-NN untuk setiap *N-gram*. Tahap-tahap yang diperlukan ketika menerapkan metode K-NN adalah sebagai berikut.

a. Menentukan parameter *k*, yakni dengan mengikuti langkah pada tahap 1 di subbab 2.9.
b. Menghitung jarak *euclidean* dengan menggunakan persamaan (2.14).
c. Klasifikasi sentimen positif dan negatif berdasarkan kelas yang memiliki jumlah anggota terbanyak di *k* tetangga terdekat.

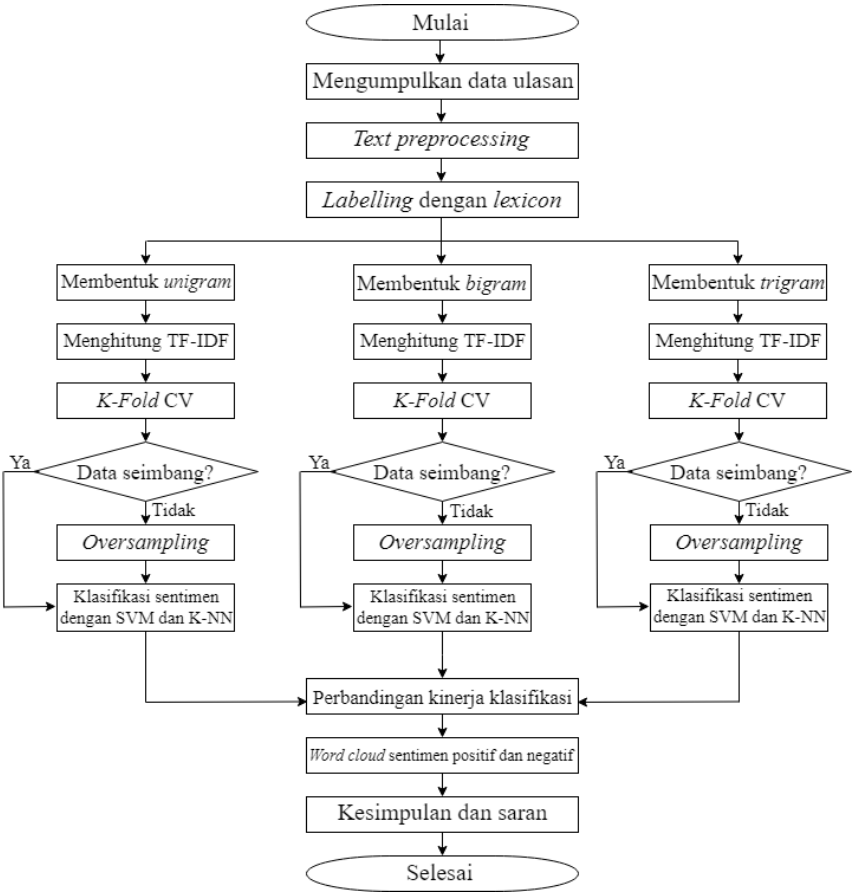
10. Klasifikasi sentimen wisatawan Candi Borobudur menggunakan metode SVM untuk setiap *N-gram*, dengan tahap-tahap berikut ini.

a. Menentukan parameter *C* untuk *kernel Linear*, serta parameter *C* dan γ untuk *kenel* RBF.
   Parameter *C* dan γ diuji coba menggunakan angka yang berbeda hingga diperoleh kinerja klasifikasi yang optimum.
b. Membuat model SVM berdasarkan jenis *kernel* dan *N-gram* terbaik. Fungsi *kernel Linear* dan RBF ditampilkan pada Tabel 2.3

11. Evaluasi kinerja klasifikasi.
    Membandingkan kinerja klasifikasi antara metode KNN dengan SVM *kernel Linear* dan RBF berdasarkan rata-rata nilai akurasi, sensitivitas, spesifisitas, dan AUC dengan menerapkan persamaan (2.15) hingga (2.18). Metode klasifikasi terbaik dipilih berdasarkan nilai AUC tertinggi.
12. Memvisualisasikan sentimen positif dan negatif dengan *word cloud* berdasarkan jenis *N-gram* terbaik. Melalui *word cloud*, dapat diperoleh informasi mengenai kata yang sering muncul pada kelas positif dan negatif.
13. Interpretasi, menarik kesimpulan, serta saran.

3.4 Diagram Alir

Diagram alir dari langkah analisis data ini ditampilkan pada Gambar 3.1.



Gambar 3.1 Diagram Alir Penelitian

(Halaman ini sengaja dikosongkan)

BAB IV
ANALISIS DAN PEMBAHASAN

Pada penelitian ini dilakukan klasifikasi sentimen positif dan negatif berdasarkan ulasan yang diberikan oleh wisatawan Candi Borobudur melalui *website TripAdvisor*. Data ulasan tersebut diperoleh melalui teknik *web scraping*, sehingga dihasilkan data seperti pada Lampiran 1. Terdapat dua metode klasifikasi yang digunakan, yakni *K-Nearest Neighbor* (K-NN) dan *Support Vector Machine* (SVM). Di kedua metode klasifikasi tersebut juga diterapkan teknik *N-gram*, yaitu menggabungkan beberapa kata sebanyak *n* yang saling berdekatan secara bersamaan (Koehn, 2009). Pada penelitian ini terdapat dugaan bahwa terjadi kasus *imbalance*, yakni tidak seimbangnya jumlah data di kelas positif dan negatif. Maka dari itu di setiap metode klasifikasi dilakukan pembuatan data *synthetic* (*oversampling*) di data *training* dengan metode *Synthetic Minority Oversampling* (SMOTE). Kemudian untuk menentukan metode klasifikasi terbaik, maka dilakukan perbandingan hasil klasifikasi setiap metode berdasarkan nilai *Area Under Curve* (AUC) di data *testing* (Bekkar, Djemaa, & Alitouche, 2013). Selain itu visualisasi menggunakan *word cloud* juga dilakukan terhadap ukuran *N-gram* yang terbaik. Dengan demikian dapat diketahui kata yang sering muncul di sentimen positif dan negatif.

4.1 Klasifikasi Sentimen dengan Metode *K-Nearest Neighbor* (K-NN) dan *Support Vector Machine* (SVM)

Sebelum melakukan analisis pada klasifikasi sentimen, terdapat beberapa tahap yang dilakukan terhadap data ulasan. Tahap-tahap tersebut terdiri dari *web scraping*, *text preprocessing*, *labelling*, membentuk *N-gram*, melakukan analisis karakteristik data ulasan, mendapatkan nilai bobot TF-IDF (*term weighting*), mempartisi data menjadi *training* dan *testing*, serta *oversampling* pada data *training* untuk mengatasi dugaan kasus data *imbalance*.

Teknik *web scraping* dilakukan dalam penelitian ini untuk memperoleh data ulasan wisatawan Candi Borobudur pada *website TripAdvisor*. Sebelum melakukan teknik *web scraping*, diperlukan

menulis *keyword* "*Borobudur temple*" pada bagian *search* yang terdapat di situs *TripAdvisor*. Sehingga akan diperoleh halaman *review* wisata Candi Borobudur seperti Gambar 4.1.

Gambar 4.1 Halaman *Review* Wisata Candi Borobudur di Situs *TripAdvisor* (Sumber: *tripadvisor.com*)

Dalam penelitian ini, elemen *website TripAdvisor* yang dibutuhkan informasinya hanyalah ulasan. Setelah didapatkan informasi yang dibutuhkan, yakni sebanyak 3591 ulasan, hasil *scraping* tersebut disimpan pada suatu *database*. Beberapa data ulasan yang diperoleh dari *scraping* terhadap *website TripAdvisor* ditampilkan pada Lampiran 1.

Data ulasan wisatawan Candi Borobudur yang telah terkumpul kemudian dilakukan *text preprocessing*. Hal tersebut bertujuan untuk menghilangkan beberapa unsur yang tidak diperlukan dalam tahap analisis, seperti tanda baca, angka, dan kata penghubung atau keterangan. *Text preprocessing* dalam penelitian ini meliputi *case folding*, yaitu mengubah seluruh teks menjadi huruf kecil (non kapital), serta menghilangkan angka dan tanda baca. Kemudian diterapkan *lemmatization*, yaitu proses mengembalikan suatu kata ke bentuk normalnya (*lemma*). Selanjutnya *stopwords removal*, yakni menghapus kata pada data ulasan yang teridentifikasi dalam daftar *stopwords*. Kemudian tahap terakhir adalah *tokenizing*, yakni pemecahan kalimat ulasan menjadi kata per kata. Hasil *text preprocessing* terhadap salah satu data ulasan ditampilkan pada Tabel 4.1. Salah satu kalimat ulasan yang digunakan untuk ilustrasi *text preprocessing* adalah "*Amazing place, sunrise was great, but it is worth to spend time to see the temple properly with a guide to understand the complexity of the place and the work done.*".

Beberapa data ulasan hasil *text preprocessing* ditampilkan pada Lampiran 4.

Tabel 4.1 Ilustrasi *Text Preprocessing*

No.	Tahap	Hasil	Keterangan
1	*Case Folding*	*amazing place sunrise was great but it is worth to spend time to see the temple properly with a guide to understand the complexity of the place and the work done*	Mengubah huruf kapital menjadi huruf kecil (non kapital) pada kata "*Amazing*", serta menghapus tanda baca koma (,) dan titik (.)
2	*Lemmatization*	*amaze place sunrise be great but it be worth to spend time to see the temple properly with a guide to understand the complexity of the place and the work do*	*amazing → amaze* *was → be* *done → do*
3	*Stopword Removal*	*amaze sunrise great worth spend properly guide understand complexity work*	Menghapus kata *place, be, but, it, to, time, see, the, temple, with, a, do* karena teridentifikasi dalam *stopwords list*
4	*Tokenizing*	"*amaze*" "*sunrise*" "*great*" "*worth*" "*spend*" "*properly*" "*guide*" "*understand*" "*complexity*" "*work*"	Pemecahan kalimat ulasan berdasarkan kata-kata yang menyusunnya

Hasil dari *text preprocessing* adalah terbentuknya struktur data, dengan variabel penelitian (*feature*) berupa kata. Struktur data yang diperoleh setelah *text preprocessing* ditampilkan pada Tabel 4.2.

Tabel 4.2 Ilustrasi Struktur Data setelah *Text Preprocessing*

Ulasan	Kata (*Feature*)						
	absolute	***...***	***great***	***...***	***visit***	***...***	***wrong***
1	0	...	0	...	0	...	0
2	0	...	0	...	1	...	0

Tabel 4.2 Ilustrasi Struktur Data setelah *Text Preprocessing* (lanjutan)

Ulasan	**Kata (*Feature*)**						
	absolute	**...**	***great***	**...**	***visit***	**...**	***wrong***
3	0	...	0	...	0	...	0
⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
6	0		1		0		0
⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
3590	0	...	0	...	1	...	0
3591	0	...	0	...	0	...	0

Adanya *text preprocessing* mengakibatkan berkurangnya jumlah *feature*, yakni menjadi 783 *feature*. Hal tersebut dikarenakan hilangnya kata-kata yang tidak bermakna signifikan.

Data ulasan wisatawan Candi Borobudur yang telah terkumpul belum dibedakan menjadi sentimen positif atau negatif. Oleh karena itu, dilakukan *labelling* dengan *lexicon* untuk mendapatkan sentimen dari setiap ulasan. Dengan menerapkan persamaan (2.1), maka akan diperoleh jenis sentimen suatu ulasan yang diilustrasikan pada Tabel 4.3.

Tabel 4.3 Ilustrasi Hasil *Labelling* menggunakan *Lexicon*

Hasil *Text Preprocessing*	**Skor**	**Label Sentimen**
amaze *sunrise* ***great worth*** *spend* ***properly*** *guide understand complexity work*	4 – 0 = 4	Positif
sunrise ***crowd*** *largest heritage* ***impress*** *scale* ***masterpiece***	2 – 1 = 1	Positif
absolutely ***overprice*** *entrance public bus accommodation approximate cost visit*	0 – 1 = -1	Negatif

Kata yang ditandai dengan huruf tebal berarti bahwa kata tersebut memiliki makna positif dalam kamus *lexicon*. Sedangkan untuk makna negatif, ditandai dengan huruf tebal dan memiliki garis bawah. Sesuai dengan persamaan (2.1), maka suatu ulasan akan diberi label sentimen positif ketika skor > 0 dan negatif jika skor < 0. Dalam penelitian ini, dilakukan peninjauan ulang untuk menentukan jenis sentimen suatu ulasan ketika skor *labelling* yang dihasil-

kan adalah 0 atau netral. Jika dalam sentimen netral tidak teridentifikasi kata sentimen positif atau negatif, maka akan diberi label positif. Data tersebut tetap dilibatkan dalam analisis dengan pertimbangan bahwa kemungkinan terdapat informasi tambahan yang berguna dalam data tersebut. Sedangkan jika sentimen netral teridentifikasi memiliki jumlah kata positif dan negatif yang sama, maka label yang diberikan adalah negatif. Pertimbangan dilakukannya hal tersebut adalah sentimen negatif akan lebih informatif (Mulajati & Hakim, 2017), sehingga pihak pengelola wisata Candi Borobudur dapat melakukan evaluasi berdasarkan keluhan wisatawan. Beberapa hasil tahap *labelling* pada data ulasan wisatawan disajikan pada Lampiran 6.

Selanjutnya diterapkan teknik *N-gram* yang bertujuan untuk memperoleh informasi yang lebih informatif. Selain itu dilihat pula pengaruh *N-gram* terhadap metode K-NN dan SVM. Dengan menerapkan *N-gram*, diharapkan dapat meningkatkan kinerja suatu *classifier*. Jenis *N-gram* yang digunakan pada penelitian ini adalah *unigram*, *bigram*, dan *trigram*. Kata-kata (*feature*) yang terbentuk dari *unigram* sama dengan hasil *text preprocessing*, yakni berjumlah 783 *feature*, dengan struktur data seperti yang telah diilustrasikan pada Tabel 4.2. Sedangkan untuk *bigram* dan *trigram*, *feature* yang terbentuk adalah gabungan dari dua dan tiga kata berurutan berdasarkan hasil *text preprocessing*. Ilustrasi struktur data *bigram* dan *trigram* disajikan pada Tabel 4.4 dan Tabel 4.5.

Tabel 4.4 Ilustrasi Struktur Data *Bigram*

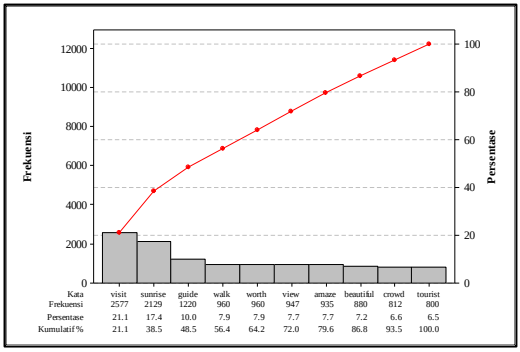
Ulasan	**Kata (*Feature*)**						
	amaze architecture	**...**	***enjoy visit***	**...**	***heritage site***	**...**	***worth visit***
1	0	...	0	...	1	...	0
2	0	...	0	...	0	...	0
3	0	...	0	...	0	...	0
⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
6	0	...	0	...	0	...	0
⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
3590	0	...	0	...	0	...	0
3591	0	...	0	...	0	...	0

Jumlah *feature* yang terbentuk pada *bigram* menjadi jauh lebih sedikit dibandingkan dengan *unigram*, yakni sebanyak 218 *feature*. Sedangkan untuk *trigram* terbentuk 151 *feature*, dimana jumlah ini tidak berbeda jauh dengan *bigram*.

Tabel 4.5 Ilustrasi Struktur Data *Trigram*

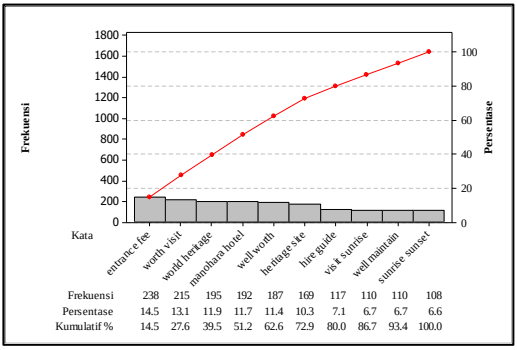
Ulasan	**Kata (*Feature*)**						
	absolutely worth visit	**...**	***combine ticket prambanan***	**...**	***pay entrance fee***	**...**	***worth visit visit***
1	0	...	1	...	0	...	0
2	0	...	1	...	0	...	0
⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
6	0	...	0	...	0	...	0
⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
3590	0	...	0	...	0	...	0
3591	0	...	0	...	0	...	0

Setelah diperoleh data ulasan yang terstruktur dengan baik, maka tahap selanjutnya yang dapat dilakukan adalah menganalisis karakteristik data ulasan wisatawan Candi Borobudur. Karakteristik data tersebut ditampilkan dalam bentuk *pareto chart*, sehingga diketahui kontribusi suatu kata dalam sekumpulan ulasan. Gambar 4.2 menunjukkan *pareto chart* pada *unigram*.

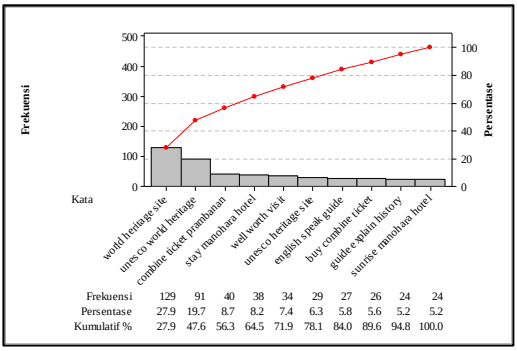


Gambar 4.2 *Pareto Chart Unigram*

Dari sepuluh kata yang terdapat dalam Gambar 4.2, kata ke-7 telah memiliki persentase kumulatif sebesar 80%. Artinya, kata "*visit*", "*sunrise*", "*guide*", "*walk*", "*worth*", "*view*", dan "*amaze*" merupakan kata-kata yang memiliki kontribusi tinggi dalam kumpulan ulasan wisatawan Candi Borobudur. Adapun Gambar 4.3 merupakan *pareto chart* untuk *bigram* dan *trigram*.



a)

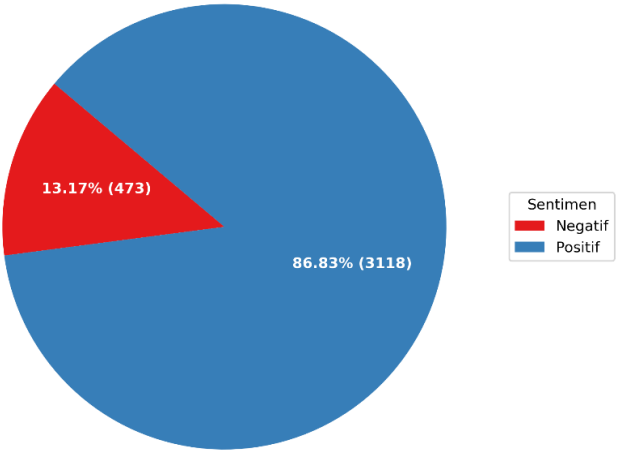


b)

Gambar 4.3 *Pareto Chart* a) *Bigram* dan b) *Trigram*

Untuk *bigram*, kata-kata yang berkontribusi tinggi terhadap ulasan adalah "*entrance fee*", "*worth visit*", "*world heritage*", "*manohara hotel*", "*well worth*", "*heritage site*", dan "*hire guide*". Sedangkan untuk *trigram*, "*world heritage site*", "*unesco world heritage*", "*combine ticket prambanan*", "*stay manohara hotel*", "*well worth*

visit", "*unesco heritage site*", dan "*english speak guide*" memiliki kontribusi yang tingga pada 3591 data ulasan yang ada. Hal tersebut didasarkan pada persentase kumulatif yang telah bernilai 80% di kata ke-7. Secara keseluruhan, hal yang kerap dibahas oleh wisatawan dalam ulasannya adalah mengunjungi Candi Borobudur ketika matahari terbit (*sunrise tour*) dengan akses melalui Hotel Manohara, menyewa pemandu wisata, dan tiket masuk ke Candi Borobudur. Selain itu banyak pula wisatawan yang mengakui bahwa Candi Borobudur merupakan salah satu wisata warisan dunia, sehingga layak untuk dikunjungi.



Gambar 4.4 Persentase Sentimen Positif dan Negatif

Hasil *labelling* dengan *lexicon* yang divisualisasikan dalam bentuk *pie chart* dalam Gambar 4.4 menunjukkan bahwa banyak wisatawan yang menulis ulasan positif mengenai tempat wisata Candi Borobudur. Ulasan bersentimen positif memiliki persentase sebesar 86,83%. Sedangkan sisanya, yaitu sebanyak 13,17%, merupakan sentimen negatif.

Tahap selanjutnya adalah *term weighting*, yaitu suatu proses untuk memperoleh nilai bobot suatu kata di dalam dokumen. Metode *term weighting* yang digunakan dalam penelitian ini adalah TF-IDF. Untuk memperoleh nilai bobot TF-IDF, terlebih dahulu dila-

kukan perhitungan *Term Frequency* (TF). Kemudian dilanjutkan dengan menghitung *Document Frequency* (DF) serta *Inverse Document Frequency* (IDF) dari setiap kata menggunakan persamaan (2.2). Ilustrasi perhitungan TF dan IDF untuk *unigram* ditampilkan pada Tabel 4.6. Sedangkan ilustrasi hasil perhitungan TF-IDF untuk *unigram* terdapat pada Tabel 4.7.

Tabel 4.6 Ilustrasi Perhitungan TF dan IDF *Unigram*

Kata	**Ulasan**							**DF**	**IDF**
	1	**2**	**...**	**6**	**...**	**3590**	**3591**		
absolute	0	0	...	0	...	0	0	29	$log\left(\frac{3591}{29}\right) = 2{,}09$
⋮	⋮	⋮	⋱	⋮	⋱	⋮	⋮	⋮	⋮
great	0	0	...	1	...	0	0	585	$log\left(\frac{3591}{585}\right) = 0{,}79$
⋮	⋮	⋮	⋱	⋮	⋱	⋮	⋮	⋮	⋮
visit	0	1	...	0	...	1	0	2577	$log\left(\frac{3591}{2577}\right) = 0{,}14$
⋮	⋮	⋮	⋱	⋮	⋱	⋮	⋮	⋮	⋮
wrong	0	0	...	0	...	0	0	42	$log\left(\frac{3591}{42}\right) = 1{,}93$

Dari empat contoh kata yang terdapat pada Tabel 4.6, kata "*visit*" memiliki nilai IDF yang paling rendah dibandingkan tiga kata lainnya. Hal tersebut berarti bahwa "*visit*" merupakan jenis kata yang muncul di hampir seluruh ulasan, dimana dapat diidentifikasi dari nilai DF sebesar 2577. Sedangkan kata "*absolute*" yang memiliki nilai IDF tertinggi di antara ketiga contoh kata lainnya menunjukkan bahwa kata tersebut hanya muncul beberapa kali di sejumlah data ulasan.

Tabel 4.7 Ilustrasi Hasil TF-IDF (w_{ij}) *Unigram*

Kata	**Ulasan**						
	1	**2**	**...**	**6**	**...**	**3590**	**3591**
absolute	0	0	...	0	...	0	0
⋮	⋮	⋮	⋱	⋮	⋱	⋮	⋮

Tabel 4.7 Ilustrasi Hasil TF-IDF (w_{ij}) *Unigram* (lanjutan)

Kata	Ulasan						
	1	**2**	**...**	**6**	**...**	**3590**	**3591**
great	0	0	...	0,79	...	0	0
⋮	⋮	⋮	⋱	⋮	⋱	⋮	⋮
visit	0	0,14	...	0	...	0,14	0
⋮	⋮	⋮	⋱	⋮	⋱	⋮	⋮
wrong	0	0	...	0	...	0	0

Nilai TF-IDF diperoleh setelah diketahui nilai DF dan IDF yang diilustrasikan dalam Tabel 4.6, sehingga dapat dilakukan perhitungan TF-IDF dengan cara mensubstitusikan kedua nilai tersebut pada persamaan (2.2). Perhitungan nilai bobot TF-IDF untuk *feature* pada *bigram* dan *trigram* memiliki cara yang sama dengan ilustrasi *unigram*. Hasil pembobotan TF-IDF untuk *unigram*, *bigram*, dan *trigram* ditampilkan pada Lampiran 9, Lampiran 10, dan Lampiran 11. Hasil pembobotan tersebut kemudian digunakan sebagai variabel prediktor untuk tahap klasifikasi sentimen.

Tahap terakhir yang dilakukan sebelum melakukan klasifikasi sentimen adalah mempartisi data *training* dan *testing* menggunakan metode 10-*fold* CV. Gambar 4.4 menunjukkan jauhnya perbedaan antara jumlah sentimen positif dan negatif, yang menjadi indikasi bahwa dalam penelitian ini terjadi kasus data *imbalance*. Oleh karena itu, diterapkan *oversampling* menggunakan SMOTE pada data *training* untuk setiap metode klasifikasi. Dalam penelitian ini, tidak dilakukan perbandingan kinerja klasifikasi antara penerapan SMOTE dan tanpa penerapan SMOTE. Tabel 4.8 merupakan ilustrasi penerapan metode SMOTE pada prosedur 10-*fold* CV untuk *fold* pertama.

Tabel 4.8 Ilustrasi SMOTE pada *Fold* Pertama

Fold 1	100% Data ulasan (3591)	
	90% *Training* (3232)	10% *Testing* (359)
Awal	Positif: 2803 Negatif: 429	Positif: 315 Negatif: 44
SMOTE	Positif: 2803 Negatif: 2803	-

Jumlah ulasan yang digunakan pada penelitian ini adalah 3591 data. Dimisalkan jumlah data *training* yang terbentuk pada *fold* 1 adalah 3232 data, dengan rincian terdapat 2803 ulasan bersentimen positif dan 429 lainnya bersentimen negatif. Maka SMOTE akan melakukan *oversampling* di kelas minor (negatif) sebanyak 2374 data. Hal tersebut mengakibatkan jumlah data di kelas positif dan negatif pada data *training fold* pertama menjadi seimbang, yakni 2803 data di masing-masing kelas. Meskipun dalam penelitian ini diterapkan SMOTE, namun SMOTE hanya dilakukan di data *training*. Sedangkan untuk data *testing* tetap mengalami *imbalance*, sehingga nantinya digunakan AUC sebagai ukuran evaluasi dari suatu metode klasifikasi.

Setelah melalui tahap *text preprocessing* hingga *oversampling*, maka seluruh data ulasan yang ada dapat digunakan untuk melakukan klasifikasi sentimen menggunakan metode K-NN dan SVM (*kernel Linear* dan RBF).

4.1.1 Klasifikasi menggunakan *K-Nearest Neighbor* (K-NN)

Pada metode K-NN, langkah awal yang perlu dilakukan yakni mencari jarak terdekat antara data *testing* dengan *training* sejumlah *k*. Untuk memudahkan pemahaman mengenai cara kerja dari metode K-NN, maka dilakukan ilustrasi menggunakan data yang ditampilkan pada Tabel 4.9.

Tabel 4.9 Ilustrasi Data

Ulasan	*beautiful*	*crowd*	*facility*	*fee*	*guide*	Sentimen
1	0	0	0,21	0	0,25	Positif
2	0	0	0	0	0,40	Positif
3	0,14	0,15	0,0	0	0	Positif
4	0	0,14	0	0	0	Negatif
5	0	0	0,0	0,18	0	Negatif
6	0	0	0,0	0	0,27	?

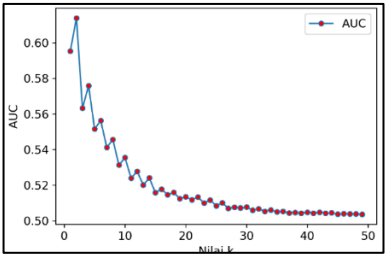
Tabel 4.9 menunjukkan bahwa data ke-6 (data *testing*) belum memiliki sentimen. Untuk menentukan kelas sentimen pada data tersebut, maka dilakukan perhitungan jarak terdekat antara data *testing* dengan masing-masing data *training* yang diilustrasikan pada

Tabel 4.10. Metode jarak terdekat yang digunakan dalam penelitian ini adalah *euclidean*, yang dirumuskan pada persamaan (2.14).

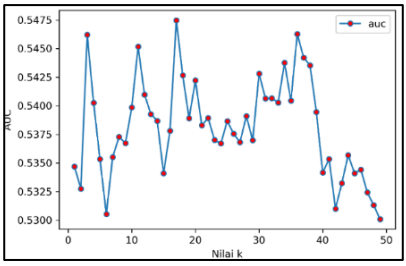
Tabel 4.10 Ilustrasi Perhitungan Jarak *Euclidean*

Ulasan	*beautiful*	*crowd*	*facility*	*fee*	*guide*	Sentimen	Jarak
1	**0**	**0**	**0,21**	**0**	**0,25**	**Positif**	**0,21**
2	**0**	**0**	**0**	**0**	**0,40**	**Positif**	**0,13**
3	0,14	0,15	0,0	0	0	Positif	0,34
4	**0**	**0,14**	**0**	**0**	**0**	**Negatif**	**0,30**
5	0	0	0,0	0,18	0	Negatif	0,32

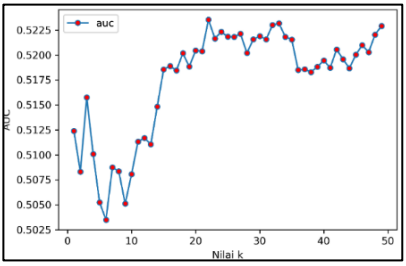
Setelah jarak antara data *testing* dengan setiap data *training* diketahui, selanjutnya nilai-nilai tersebut diurutkan dari yang terendah hingga tertinggi dan hanya memilih *k* tetangga terdekat. Dimisalkan nilai *k* = 3, maka dapat diketahui tetangga terdekat dari data *testing* adalah data ke-1, ke-2, dan ke-5. Dari ketiga data tersebut, dua diantaranya memiliki label positif. Sehingga data *testing* akan diklasifikasikan ke dalam kelas positif.



a)



b)



c)

Gambar 4.5 Perbandingan Nilai *k* (a) *Unigram*, (b) *Bigram*, dan (c) *Trigram* terhadap Metode K-NN

Dalam metode K-NN, tidak terdapat angka pasti mengenai nilai *k* yang harus digunakan. Sehingga dalam penelitian ini digunakan nilai *k* yang dimulai dari 1 hingga 50 (Wang, Neskovic, & Cooper, 2007) pada setiap jenis *N-gram*. Hal tersebut bertujuan untuk melihat pengaruh nilai *k* terhadap kinerja klasifikasi, yaitu rata-rata nilai AUC dari 10-*fold* CV di data *testing*. Hasil kinerja klasifikasi untuk setiap nilai *k* pada *unigram*, *bigram*, dan *trigram* divisualisasikan melalui Gambar 4.3.

Gambar 4.3 menunjukkan bahwa untuk *unigram*, nilai $k = 2$ memiliki kinerja klasifikasi yang paling optimum. Rata-rata AUC yang dihasilkan ketika nilai $k = 2$ adalah sebesar 0,6139. Sedangkan nilai *k* yang optimum untuk *bigram* dan *trigram* adalah 17 dan 22, dengan rata-rata AUC sebesar 0,5475 dan 0,5235. Adapun hasil kinerja klasifikasi dari *unigram*, *bigram*, dan *trigram* untuk 10 *fold* menggunakan nilai *k* terbaik ditampilkan pada Tabel 4.11.

Tabel 4.11 Kinerja Klasifikasi K-NN setiap *Fold*

N-gram	***Fold***	**Akurasi**	**Spesifisitas**	**Sensitivitas**	**AUC**
Unigram $k = 2$	1	0,41	0,33	0,92	0,6261
	2	0,46	0,39	0,85	0,6185
	3	0,42	0,35	0,95	0,6497
	4	0,41	0,34	0,88	0,6100
	5	0,41	0,34	0,95	0,6455
	6	0,40	0,33	0,85	0,5943
	7	0,37	0,30	0,86	0,5800
	8	0,43	0,37	0,86	0,6144
	9	0,40	0,33	0,88	0,6034
	10	0,39	0,31	0,88	0,5970
Rata-rata		**0,41**	**0,34**	**0,89**	**0,6139**
Bigram $k = 17$	1	0,49	0,46	0,63	0,5452
	2	0,49	0,47	0,62	0,5406
	3	0,48	0,45	0,63	0,5437
	4	0,47	0,46	0,55	0,5054
	5	0,45	0,45	0,50	0,4740
	6	0,49	0,46	0,67	0,5658
	7	0,46	0,44	0,59	0,5165
	8	0,51	0,49	0,73	0,6087

Tabel 4.11 Kinerja Klasifikasi KNN setiap *Fold* (lanjutan)

N-gram	*Fold*	Akurasi	Spesifisitas	Sensitivitas	AUC
Bigram	9	0,52	0,49	0,69	0,5897
k = 17	10	0,48	0,44	0,73	0,5851
Rata-rata		**0,48**	**0,46**	**0,63**	**0,5475**
	1	0,82	0,94	0,12	0,5294
	2	0,82	0,95	0,08	0,5140
	3	0,85	0,96	0,12	0,5419
Trigram	4	0,84	0,95	0,12	0,5352
k = 22	5	0,81	0,94	0,04	0,4882
	6	0,84	0,94	0,10	0,5187
	7	0,80	0,91	0,11	0,5083
	8	0,78	0,89	0,13	0,5136
	9	0,84	0,93	0,12	0,5249
	10	0,85	0,94	0,19	0,5614
Rata-rata		**0,82**	**0,93**	**0,11**	**0,5235**

Hasil kinerja klasifikasi dari 10 *fold* menunjukkan bahwa untuk metode K-NN, *N-gram* dengan nilai n = 1 (*unigram*) memiliki kinerja yang lebih baik dibandingkan *bigram* dan *trigram*. Hal tersebut didasarkan pada rata-rata AUC sebesar 0,6139. Apabila diinterpretasikan lebih lanjut, kinerja klasifikasi yang dihasilkan oleh metode K-NN dengan hanya menerapkan *unigram* telah tergolong cukup atau *fair*. Spesifisitas yang memiliki nilai rata-rata sebesar 0,34 memiliki arti bahwa kemampuan metode K-NN untuk mengklasifikasikan sentimen positif dengan tepat hanya sebesar 34%. Namun metode K-NN dapat mengklasifikasikan sentimen negatif dengan persentase ketepatan sebesar 89%. Untuk menunjukkan jumlah data yang diklasifikasikan dengan tepat dan salah terklasifikasi (misklasifikasi), maka pada Lampiran 13, Lampiran 14, dan Lampiran 15 ditampilkan tabel *confusion matrix* metode K-NN untuk setiap jenis *N-gram*.

4.1.2 Klasifikasi menggunakan Metode *Support Vector Machine* (SVM)

Metode SVM sering digunakan untuk menyelesaikan kasus klasifikasi karena memiliki kinerja yang baik. SVM memiliki berbagai jenis *kernel* yang dapat digunakan untuk menangani kasus *li-*

nearly dan *non-linearly separable data*, yakni *Linear*, RBF, *Polynomial*, *Sigmoid*, dan *Inverse Multiquadric Function*. Dari lima jenis *kernel* yang ada, hanya dua *kernel* yang digunakan pada penelitian ini, yakni *Linear* dan RBF.

SVM *kernel Linear* memiliki suatu parameter, yaitu *cost* (*C*). Untuk menentukan nilai *C* terbaik yang dapat digunakan memprediksi data *testing* secara akurat, maka pada setiap jenis *N-gram* dilakukan uji coba menggunakan nilai *C* yang berbeda-beda. Dalam penelitian ini, pemilihan nilai *C* dilakukan berdasarkan nilai AUC tertinggi pada data *testing* karena kinerja klasifikasi yang tinggi pada data *training* justru tidak memiliki arti. Hal tersebut disebabkan *classifier* dapat memprediksi data *training* dengan baik karena label data *training* telah diketahui (Hsu, Chang, & Lin, 2003).

Umumnya rentang nilai *C* yang digunakan ketika *trial and error* adalah 10^{-2} hingga 10^{4} (Huang dkk., 2007). Namun rentang nilai *C* yang digunakan pada penelitian ini adalah 10^{-2} hingga 10^{2} dengan pertimbangan efisiensi waktu. Perbandingan kinerja klasifikasi berdasarkan AUC di setiap nilai *C* untuk *unigram*, *bigram*, dan *trigram* ditampilkan pada Tabel 4.12. Terlihat bahwa nilai *C* optimum untuk *unigram*, *bi-gram*, dan *trigram* secara berturut-turut adalah 10^{0}, 10^{-2}, dan 10^{0}. Nilai AUC yang terdapat pada tabel tersebut merupakan nilai rata-rata AUC data *testing* dari hasil pembagian data *training-testing* menggunakan 10-*fold* CV.

Tabel 4.12 Perbandingan Nilai *C* terhadap Kinerja SVM *Kernel Linear*

N-gram	***C***				
	10^{-2}	**10^{-1}**	**10^{0}**	**10^{1}**	**10^{2}**
Unigram	0,7551	0,8142	**0,8143**	0,7673	0,7514
Bigram	**0,6009**	0,6001	0,5745	0,5842	0,5840
Trigram	0,5184	0,5331	**0,5335**	0,5268	0,5266

Berdasarkan nilai *C* yang optimum, da-pat diketahui rincian kinerja klasifikasi dari seluruh *fold* yang disa-jikan pada Tabel 4.13. Dari tiga jenis *N-gram* yang digunakan, SVM *kernel Linear* yang memiliki kinerja klasifikasi paling optimum adalah ketika menggunakan *unigram*. Rata-rata nilai AUC yang dihasilkan metode ini mampu

mencapai 0,8143, dimana nilai tersebut tergolong memiliki kinerja klasifikasi yang sangat baik (Bekkar, Djemaa, & Alitouche, 2013). Secara rata-rata, sebesar 90% sentimen positif terklasifikasikan dengan tepat ketika menggunakan *unigram*. Persentase tersebut dilihat dari nilai spesifisitas. Sedangkan untuk sentimen negatif, persentase kemampuan yang dihasilkan pada metode ini lebih rendah dibandingkan kelas positif, yakni sebesar 73%. Tabel *confusion matrix* SVM *kernel Linear* untuk setiap *N-gram* ditampilkan pada Lampiran 16, Lampiran 17, dan Lampiran 18.

Tabel 4.13 Kinerja Klasifikasi SVM *Kernel Linear* setiap *Fold*

N-gram	***Fold***	**Akurasi**	**Spesifisitas**	**Sensitivitas**	**AUC**
Unigram $C = 1$	1	0,87	0,89	0,73	0,8095
	2	0,87	0,90	0,75	0,8206
	3	0,90	0,93	0,68	0,8044
	4	0,89	0,89	0,84	0,8665
	5	0,86	0,90	0,57	0,7331
	6	0,88	0,90	0,77	0,8356
	7	0,90	0,92	0,75	0,8337
	8	0,89	0,90	0,78	0,8394
	9	0,87	0,88	0,79	0,8351
	10	0,85	0,88	0,65	0,7652
Rata-rata		**0,88**	**0,90**	**0,73**	**0,8143**
Bigram $C = 0{,}01$	1	0,84	0,93	0,24	0,5873
	2	0,80	0,88	0,35	0,6148
	3	0,82	0,91	0,20	0,5562
	4	0,81	0,87	0,34	0,6070
	5	0,60	0,60	0,59	0,5954
	6	0,82	0,90	0,23	0,5660
	7	0,79	0,85	0,39	0,6189
	8	0,81	0,89	0,31	0,5982
	9	0,59	0,57	0,70	0,6340
	10	0,79	0,85	0,41	0,6315
Rata-rata		**0,77**	**0,82**	**0,38**	**0,6009**
Trigram $C = 1$	1	0,29	0,21	0,91	0,5587
	2	0,28	0,18	0,90	0,5419
	3	0,30	0,23	0,82	0,5234

Tabel 4.13 Kinerja Klasifikasi SVM *Kernel Linear* setiap *Fold* (lanjutan)

N-gram	*Fold*	Akurasi	Spesifisitas	Sensitivitas	AUC
Trigram $C = 1$	4	0,28	0,20	0,84	0,5220
	5	0,27	0,19	0,80	0,4980
	6	0,30	0,22	0,84	0,5316
	7	0,27	0,18	0,91	0,5434
	8	0,32	0,22	0,94	0,5807
	9	0,31	0,21	0,84	0,5237
	10	0,30	0,21	0,82	0,5114
Rata-rata		**0,29**	**0,20**	**0,86**	**0,5335**

Berbeda dengan *kernel Linear*, terdapat dua parameter yang dimiliki oleh SVM *kernel* RBF, yaitu *C* dan *gamma* (γ). Hal yang perlu diperhatikan pada *kernel* RBF adalah penentuan kombinasi nilai dari kedua parameter ini, karena nilai *C* dan γ yang berbeda dapat menghasilkan kinerja yang berbeda pula. Oleh sebab itu digunakan kombinasi nilai *C* dan γ yang berbeda untuk mendapatkan kombinasi parameter terbaik berdasarkan rata-rata nilai AUC 10-*fold* CV yang tertinggi pada data *testing*. Dengan menggunakan nilai $C = 10^{-2}, 10^{-1}, ..., 10^{2}$ dan $\gamma = 2^{-15}, 2^{-13}, ..., 2^{3}$ (Hsu, Chang, & Lin, 2003), maka diperoleh kombinasi nilai *C* dan γ terbaik untuk setiap jenis *N-gram* yang disajikan pada Tabel 4.14. Adapun Lampiran 19 menampilkan hasil kinerja klasifikasi SVM *kernel* RBF dari seluruh kombinasi nilai *C* dan γ.

Tabel 4.14 Kombinasi Parameter Terbaik SVM *Kernel* RBF

N-gram	Parameter		Rata-rata AUC
	C	γ	
Unigram	10^{2}	2^{-9}	0,8234
Bigram	10^{-1}	2^{-5}	0,6068
Trigram	10^{1}	2^{-5}	0,5417

Selanjutnya berbagai ukuran kinerja klasifikasi SVM *kernel* RBF yang terdiri dari akurasi, sensitivitas, spesifisitas, dan AUC dari setiap *fold* untuk masing-masing *N-gram* ditampilkan pada Tabel 4.15. Tidak berbeda dengan *kernel Linear*, pada SVM *kernel* RBF, *unigram* merupakan jenis *N-gram* dengan kinerja klasifikasi yang paling optimal. Rata-rata nilai AUC yang dihasilkan ketika

menerapkan *unigram* mampu mencapai 0,8234, sehingga dapat dikatakan memiliki kinerja yang sangat baik. Rata-rata kemampuan metode SVM *kernel* RBF dengan *unigram* untuk mengklasifikasikan sentimen positif dengan tepat bernilai sebesar 89%, sedangkan untuk sentimen negatif adalah 76%. Rincian mengenai jumlah data yang terklasifikasi dengan tepat maupun salah terklasifikasi disajikan pada Lampiran 20, Lampiran 21, dan Lampiran 22 dalam bentuk *confusion matrix*.

Tabel 4.15 Kinerja Klasifikasi SVM *Kernel* RBF setiap *Fold*

N-gram	***Fold***	**Akurasi**	**Spesifisitas**	**Sensitivitas**	**AUC**
Unigram $C = 10^{2}$ $\gamma = 2^{-9}$	1	0,86	0,86	0,84	0,8524
	2	0,87	0,90	0,75	0,8206
	3	0,89	0,92	0,70	0,8126
	4	0,87	0,88	0,86	0,8699
	5	0,87	0,90	0,61	0,7564
	6	0,87	0,89	0,75	0,8179
	7	0,90	0,91	0,82	0,8662
	8	0,89	0,90	0,80	0,8496
	9	0,85	0,87	0,77	0,8214
	10	0,84	0,86	0,67	0,7674
Rata-rata		**0,87**	**0,89**	**0,76**	**0,8234**
Bigram $C = 10^{-1}$ $\gamma = 2^{-5}$	1	0,71	0,74	0,44	0,5937
	2	0,67	0,69	0,55	0,6170
	3	0,82	0,91	0,20	0,5562
	4	0,64	0,66	0,48	0,5688
	5	0,60	0,61	0,59	0,5970
	6	0,83	0,91	0,23	0,5708
	7	0,65	0,64	0,68	0,6631
	8	0,70	0,72	0,57	0,6438
	9	0,60	0,58	0,70	0,6406
	10	0,67	0,68	0,55	0,6174
Rata-rata		**0,69**	**0,71**	**0,50**	**0,6068**
Trigram $C = 10^{1}$ $\gamma = 2^{-5}$	1	0,31	0,23	0,89	0,5587
	2	0,30	0,20	0,90	0,5516
	3	0,32	0,25	0,82	0,5329
	4	0,29	0,22	0,84	0,5300

Tabel 4.15 Kinerja Klasifikasi SVM *Kernel* RBF setiap *Fold* (lanjutan)

N-gram	*Fold*	Akurasi	Spesifisitas	Sensitivitas	AUC
Trigram $C = 10^1$ $\gamma = 2^{-5}$	5	0,27	0,19	0,83	0,5089
	6	0,31	0,25	0,77	0,5118
	7	0,29	0,21	0,91	0,5593
	8	0,34	0,25	0,96	0,6022
	9	0,33	0,24	0,84	0,5386
	10	0,31	0,23	0,82	0,5227
Rata-rata		**0,31**	**0,23**	**0,86**	**0,5417**

Hasil analisis mengenai kinerja klasifikasi metode SVM berdasarkan nilai AUC menunjukkan bahwa *kernel* RBF lebih baik dibandingkan *Linear*, yang disertai penerapan *unigram*. Dengan parameter $C = 10^2$ dan $\gamma = 2^{-9} \approx 0{,}002$, kemudian *fold* ke-4 merupakan *fold* terbaik, maka model SVM *kernel* RBF untuk *unigram* dengan menggunakan data *training* pada *fold* ke-4 dapat dituliskan ke dalam bentuk persamaan (4.1).

$$f(\mathbf{x}) = \sum_{i=1}^{2471} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) - 41{,}1822\,, \tag{4.1}$$

dengan y_i adalah kelas *support vector* (+1 atau -1), α_i menunjukkan koefisien *Lagrange*, $K(\mathbf{x}, \mathbf{x}_i) = exp(-0{,}002\,||\,\mathbf{x} - \mathbf{x}_i\,||^2)$, nilai -41,1822 menunjukkan bias (b), dan i = 1, 2, ..., 2471 merupakan banyaknya *support vector*. Persamaan lengkap model SVM dengan *kernel* RBF dan *unigram* ditampilkan pada Lampiran 23.

Persamaan (4.1) kemudian digunakan untuk mengklasifikasikan data *testing*. Dimisalkan terdapat lima data *testing*, yang ditampilkan pada Tabel 4.16.

Tabel 4.16 Contoh Data *Testing*

Data *Testing* ke-	X_1	X_2	...	X_{782}	X_{783}
1	0	0,234866	...	0	0
2	0	0	...	0	0
3	0	0	...	0	0
4	0,135936	0	...	0	0
5	0	0	...	0	0

Perhitungan $K(\mathbf{x},\mathbf{x}_i)$ pada data *testing* pertama akan menghasilkan nilai-nilai $K(\mathbf{x},\mathbf{x}_i)$ yang terdapat pada Tabel 4.17. Nilai α_i dan y_i yang telah diketahui dari persamaan (4.1) selanjutnya dikalikan dengan $K(\mathbf{x},\mathbf{x}_i)$. Sehingga diperoleh nilai $\alpha_i y_i K(\mathbf{x},\mathbf{x}_i)$ yang ditampilkan pula pada Tabel 4.17.

Tabel 4.17 Ilustrasi Perhitungan Fungsi *Hyperplane* Data *Testing* Pertama

i	α_i	y_i	$K(\mathbf{x},\mathbf{x}_i)$	$\alpha_i y_i K(\mathbf{x},\mathbf{x}_i)$
1	100	-1	0,996509	-99,6509
2	1,5593	+1	0,997579	1,555524
		⋮		
2470	100	-1	0,996188	-99,6188
2471	100	+1	0,997249	99,72486

Tabel 4.17 menunjukkan nilai $\alpha_i y_i K(\mathbf{x},\mathbf{x}_i)$ dari seluruh *i* (*support vector*), sehingga dapat dilakukan perhitungan dengan menenerapkan persamaan (4.1) yang berfungsi untuk menentukan kelas data *testing*. Ilustrasi hasil klasifikasi terhadap lima data *testing* dalam Tabel 4.16 ditampilkan pada Tabel 4.18.

Tabel 4.18 Ilustrasi Klasifikasi Lima Data *Testing*

i	$\alpha_i y_i K(\mathbf{x},\mathbf{x}_i)$				
	***Testing* 1**	***Testing* 2**	***Testing* 3**	***Testing* 4**	***Testing* 5**
1	-99,6509	-99,7264	-99,6188	-99,705	-99,6617
2	1,555524	1,5593	1,554926	1,555684	1,554656
		⋮			
2470	-99,6188	-99,7025	-99,6308	-99,696	-99,6194
2471	99,72486	99,78635	99,68228	99,75881	99,70071
$\sum_{i=1}^{2471} \alpha_i y_i K(\mathbf{x},\mathbf{x}_i)$	41,1099	42,1820	40,2745	42,0351	40,6557
$f(\mathbf{x})$	-0,0723	0,9998	-0,9077	0,8529	-0,5265
Kelas	-1	+1	-1	+1	-1
Keterangan	Positif	Negatif	Positif	Negatif	Positif

Melalui Tabel 4.18 terlihat bahwa data *testing* ke-2 dan ke-4 memiliki nilai $f(\mathbf{x}) > 0$. Hal ini menyebabkan kedua data tersebut diklasifikasikan ke dalam kelas +1 (negatif). Sedangkan untuk data

testing dengan nilai $f(\mathbf{x}) < 0$ akan terklasifikasi menjadi kelas -1 (positif).

4.2 Perbandingan Kinerja Klasifikasi Metode *K-Nearest Neighbor* (K-NN) dan *Support Vector Machine* (SVM)

Setelah diperoleh performansi klasifikasi dari setiap metode, maka dilakukan perbandingan untuk mempertimbangkan metode klasifikasi terbaik yang dapat diterapkan untuk kasus klasifikasi sentimen wisatawan Candi Borobudur. Pertimbangan tersebut dilakukan dengan membandingkan rata-rata nilai akurasi, spesifisitas, sensitivitas, dan AUC, yang ditampilkan pada Tabel 4.19.

Tabel 4.19 Performa Klasifikasi antar Metode

Metode	*N-gram*	Akurasi	Spesifisitas	Sensitivitas	AUC
SVM (RBF)	*Unigram*	0,87	0,89	0,76	0,8234
	Bigram	0,69	0,71	0,50	0,6068
	Trigram	0,31	0,23	0,86	0,5417
K-NN	*Unigram*	0,41	0,34	0,89	0,6139
	Bigram	0,48	0,46	0,63	0,5457
	Trigram	0,82	0,93	0,11	0,5235

Dari hasil perbandingan seluruh metode, kasus klasifikasi sentimen wisatawan Candi Borobudur akan lebih baik apabila diselesaikan menggunakan SVM *kernel* RBF dan teknik *N-gram* dengan $n = 1$ (*unigram*). Hal tersebut dikarenakan rata-rata nilai AUC dari metode ini mampu mencapai 0,8234, atau dengan kata lain memiliki kinerja klasifikasi sangat baik. Nilai tersebut berbeda jauh dibandingkan metode lainnya, yang hanya menghasilkan rata-rata AUC pada interval 0,50 hingga 0,61. Selain itu dapat disimpulkan pula bahwa dalam penelitian ini, bertambahnya nilai *n* pada *N-gram* justru menurunkan nilai AUC. *Unigram* menghasilkan kinerja klasifikasi yang lebih baik karena dalam penelitian ini tahap *labelling* dengan *lexicon* hanya berdasarkan satu kata saja. Kemudian hasil *labelling* tersebut digunakan untuk melakukan klasifikasi sentimen dengan teknik *unigram*, *bigram*, dan *trigram* di setiap metode klasifikasi. Adapun Tabel 4.20 menampilkan perbedaan antara hasil *labelling* berdasarkan *lexicon* dan *N-gram*, terhadap sepuluh data ulasan wi-

satawan Candi Borobudur yang telah dilakukan *text preprocessing*. *Labelling* berdasarkan *N-gram* diperoleh menurut subjektif peneliti.

Tabel 4.20 Perbedaan Hasil *Labelling* dengan *Lexicon* dan *N-gram*

No.	*Review*	*Lexicon*	*Bigram*	*Trigram*
1	*wonder world unesco world heritage site*	Positif	Positif	Positif
2	*feel visit sunrise amaze absolutely unforgettable*	Positif	Positif	Positif
3	*overcrowd statue headless mural wall partly wholly destroy rainwater collect pool damage*	Negatif	Negatif	Negatif
4	*sunrise highly recommend surround countryside life mysteriously greater force troop horde tourist*	Negatif	Positif	Positif
5	*highly recommend pay extra guide interest fact overcrowd*	Positif	Negatif	Negatif
6	*insane charge foreigner entrance close impressive angkor expensive nice wrong*	Negatif	Negatif	Negatif
7	*watch sunrise worth nice spot*	Positif	Positif	Positif
8	*walk wait sunrise door open disappoint restoration work lack history interest*	Negatif	Negatif	Negatif
9	*incredibly busy expensive visit worth visit java*	Positif	Negatif	Negatif
10	*unbelievable architecture shock visit sunrise sunset bite crowdy sunset*	Negatif	Positif	Positif

Dari sepuluh data ulasan yang dilakukan *labelling* dengan *N-gram*, empat data diantaranya diperoleh hasil yang berbeda ketika dite-

rapkan *labelling* berdasarkan *lexicon*. Adanya perbedaan hasil *labelling* antara *lexicon* dengan *N-gram* dapat mempengaruhi hasil klasifikasi ketika diterapkan teknik *unigram, bigram*, dan *trigram*.

Jumlah data yang diklasifikasikan dengan benar dari metode SVM *kernel* RBF dengan teknik *unigram* dapat diketahui melalui *confusion matrix* yang ditampilkan pada Tabel 4.21. Melalui tabel *confusion matrix* dapat diketahui bahwa total misklasifikasi ketika digunakan metode SVM *kernel* RBF dengan penerapan *unigram* adalah sebanyak 460 data. Dari 473 ulasan bersentimen negatif, 115 diantaranya diklasifikasikan menjadi sentimen positif. Dengan kata lain, persentase kesalahan klasifikasi pada kelas negatif adalah sebesar 24%. Sedangkan untuk kelas positif, persentase misklasifikasi bernilai sebesar 11%, dengan rincian terdapat 345 data ulasan diklasifikasikan bersentimen negatif.

Tabel 4.21 *Confusion Matrix* SVM *Kernel* RBF (*Unigram*)

Kelas Aktual	**Kelas Prediksi**		**Total**
	Positif	Negatif	
Positif	2773	345	**3118**
Negatif	115	358	**473**
Total	**2888**	**703**	**3591**

Selanjutnya pada Tabel 4.22 ditampilkan kinerja klasifikasi SVM *Kernel* RBF dengan teknik *unigram* pada data *training* dan *testing*.

Tabel 4.22 Perbandingan Kinerja Klasifikasi Data *Training* dan *Testing* pada SVM-RBF-*Unigram*

Data	**Akurasi**	**AUC**
Training	0,95	0,9466
Testing	0,87	0,8234

Dilakukannya perbandingan kinerja klasifikasi antara data *training* dan *testing* bertujuan untuk mengidentifikasi apakah model klasifikasi yang terbentuk telah *fit* atau dapat digunakan untuk memprediksi data-data ulasan yang masuk di waktu yang akan datang. Pada Tabel 4.22, terlihat bahwa hasil klasifikasi antara data *training* dan *testing* tidak jauh berbeda. Sehingga dapat dikatakan bahwa model

klasifikasi yang terbentuk pada data *training* tidak mengalami kasus *overfitting* maupun *underfitting*, dan model tersebut dapat digunakan untuk memprediksi sentimen wisatawan Candi Borobudur ketika terdapat ulasan yang baru.

4.3 Visualisasi Sentimen dengan *Word Cloud*

Hasil analisis pada subbab 4.2 menunjukkan bahwa jenis *N-gram* terbaik yang diterapkan untuk penelitian ini adalah *unigram*. Sehingga diperoleh Gambar 4.6 dan Gambar 4.7 yang merupakan visualisasi berupa *unigram word cloud* untuk sentimen positif dan negatif. Melalui visualisasi tersebut, dapat diketahui kata-kata yang sering muncul dalam ulasan bersentimen positif dan negatif.

Gambar 4.6 *Unigram Word Cloud* Positif

Pada *word cloud* sentimen positif, secara keseluruhan wisatawan merasa senang dan puas ketika mengunjungi wisata Candi Borobudur. Kata "*worth*", "*visit*", "*absolutely*", dan "*amaze*" yang muncul pada *word cloud* tersebut memiliki arti yakni Candi Borobudur merupakan destinasi wisata yang layak dikunjungi karena keindahannya. Selain itu wisatawan juga menganggap Candi Borobudur terawat dengan baik, yang ditandai dengan kata "*well*" dan "*clean*". Angkor Wat merupakan tempat wisata yang terletak di negara Kamboja. Munculnya kata "*angkor*" dalam Gambar 4.6 memiliki arti bahwa banyak wisatawan yang membandingkan Candi Borobudur dengan Angkor Wat. Adapun kata "*crowd*" yang sebenarnya memiliki makna negatif, namun muncul di dalam *word cloud* bersentimen positif. Hal tersebut terjadi karena kata "*crowd*" muncul beriringan dalam ulasan yang bersentimen positif, misalnya dalam

ulasan "*Took the sunrise tour. It was very crowded. But as named to be the largest Buddhist temple heritage, you will very impressed with its scale and masterpieces around the temple*."

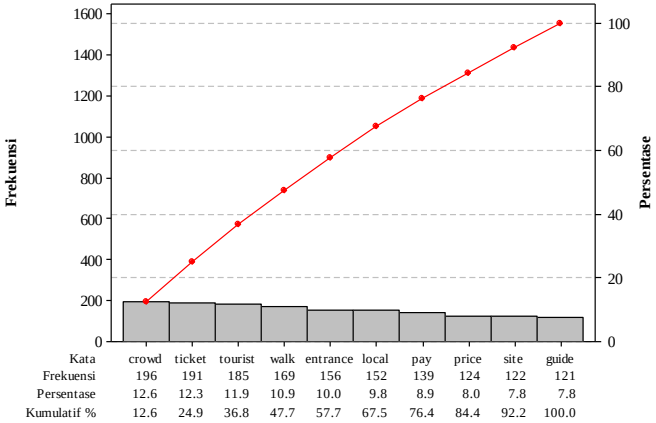
Gambar 4.7 *Unigram Word Cloud* Negatif

Dalam *word cloud* sentimen positif dan negatif, "*visit*" dan "*sunrise*" merupakan kata yang berukuran sama. Hal tersebut berarti bahwa kata "*visit*" dan "*sunrise*" memiliki frekuensi kemunculan yang sama besarnya di kedua kelas sentimen. Kedua kata tersebut juga dapat diartikan bahwa banyak wisatawan yang mengunjungi Candi Borobudur dan tertarik dengan *sunrise tour*. Wisatawan hanya dapat mengakses *sunrise tour* melalui Hotel Manohara.

Meskipun banyak wisatawan tertarik dengan program *sunrise tour*, akan tetapi tidak sedikit pula wisatawan yang mengeluhkan mahalnya harga tiket *sunrise tour*. Harga tiket untuk akses reguler pun dikeluhkan sebab dianggap mahal bagi wisatawan asing. Keluhan tersebut dapat diidentifikasi melalui adanya kata "*expensive*", "*overprice*", "*extremely*", "*entrance*", dan "*price*" pada *word cloud* sentimen negatif. Dalam Gambar 4.7 terlihat adanya kata "*amaze*" dan "*nice*", dimana kedua kata tersebut bermakna positif. Kata "*amaze*" dan "*nice*" yang muncul dalam ulasan negatif dapat menyebabkan kedua kata tersebut juga muncul dalam *word cloud* bersentimen negatif.

Untuk memperoleh informasi mengenai hal yang patut menjadi perhatian bagi pihak pengelola wisata, maka pada Gambar 4.8 ditampilkan visualisasi sepuluh kata dengan frekuensi kemunculan

tertinggi pada data ulasan bersentimen negatif dalam bentuk *pareto chart*.



Kata	crowd	ticket	tourist	walk	entrance	local	pay	price	site	guide
Frekuensi	196	191	185	169	156	152	139	124	122	121
Persentase	12.6	12.3	11.9	10.9	10.0	9.8	8.9	8.0	7.8	7.8
Kumulatif %	12.6	24.9	36.8	47.7	57.7	67.5	76.4	84.4	92.2	100.0

Gambar 4.8 *Pareto Chart* Sentimen *Negatif* (*Unigram*)

Tanpa melibatkan kata "*visit*" dan "*sunrise*" dalam *pareto chart* tersebut, dapat diperoleh informasi yakni permasalahan keramaian dan harga tiket masuk merupakan hal yang patut mendapatkan perhatian lebih. Hal ini diidentifikasi dari adanya kata "*crowd*", "*ticket*", "*entrance*", "*pay*", dan "*price*" yang muncul dalam 80% kumulatif persentase.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan yang diperoleh berdasarkan analisis dan pembahasan yang telah dilakukan adalah sebagai berikut.

1. Klasifikasi sentimen wisatawan Candi Borobudur dengan metode *K-Nearest Neighbor* memiliki kinerja yang lebih baik bila menerapkan teknik *N-gram* jenis *unigram*. Dengan metode ini, kinerja yang dihasilkan tergolong cukup atau *fair*. Sedangkan untuk metode SVM, jenis *kernel* yang terbaik adalah *Radial Basis Function* dan disertai teknik *unigram*. Kinerja klasifikasi dari metode ini berhasil mengklasifikasikan sentimen positif dan negatif dengan sangat baik. Selain itu bertambahnya ukuran *n* pada *N-gram* membuat kinerja klasifikasi menjadi menurun, baik pada metode *Support Vector Machine* maupun *K-Nearest Neighbor*.
2. Metode terbaik untuk mengklasifikasikan sentimen wisatawan Candi Borobudur adalah *Support Vector Machine kernel Radial Basis Function* yang disertai penerapan *unigram*. Rata-rata nilai AUC yang diperoleh dari metode ini mampu mencapai 0,8234. Hal tersebut menjadi indikasi bahwa metode ini memiliki kinerja klasifikasi yang sangat baik.
3. Berdasarkan hasil visualisasi *word cloud*, diperoleh informasi bahwa banyak wisatawan asing yang mengeluhkan mahalnya tiket masuk ke Candi Borobudur, baik untuk *sunrise tour* maupun reguler. Meskipun demikian, banyak wisatawan yang puas ketika berkunjung ke Candi Borobudur karena keindahannya dan kawasan yang terawat dengan baik.

5.2 Saran

Saran yang dapat diberikan untuk penelitian selanjutnya dan pihak pengelola wisata Candi Borobudur adalah sebagai berikut.

1. Untuk penelitian selanjutnya, diperlukan ketelitian pada tahap *text preprocessing* karena dapat mempengaruhi hasil klasifikasi. Dalam analisis sentimen, apabila *labelling* sentimen mengguna-

kan *lexicon* maupun *N-gram*, maka diperlukan peninjauan ulang terhadap kata-kata yang menunjukkan makna negasi, misalnya "*no*", "*not*", "*never*", "*very*", dll. Selain itu dapat juga diterapkan teknik kombinasi *unigram-bigram-trigram* dalam pembentukan *feature*.

2. Untuk pengelola wisata Candi Borobudur, sebaiknya dilakukan peninjauan ulang mengenai biaya tiket masuk reguler sehingga tidak terdapat ketimpangan antara wisatawan domestik dengan asing. Adapun hal yang perlu dipertahankan adalah tetap diadakannya program *sunrise tour* karena banyak wisatawan asing yang menyukai program tersebut.

DAFTAR PUSTAKA

Abe, S. (2010, June 16). *Support Vector Machines for Pattern Classification.* London: Springer.

Arianto, A. D., Affandi, A., & Nugroho, S. M. (2017). An Opinion Anomaly Detection Using K-Nearest Neighbours on Public Sector Financial Reports. *The 3rd International Seminar on Science and Technology, 4*(1), 1-2.

Babatunde, O. H., Diepeveen, D., Armstrong, L. J., & Leng, J. (2015). Comparative Analysis of Genetic Algorithm and Particle Swam Optimization: An Application in Precision Agriculture. *Asian Journal of Computer and Information Systems, 3*(1), 1-12.

Bekkar, M., Djemaa, H. K., & Alitouche, T. A. (2013). Evaluation Measure for Models Assessment over Imbalanced Data Sets. *Journal of Information Engineering and Aplications, 3*(10), 27-38.

Berry, M. W., & Kogan, J. (2010). *Text Mining: Applications and Theory 1st Edition.* United Kingdom: Wiley.

Borade, O., Gosavi, K., Gowda, A., & Shinde, A. (2017). Sentiment Analysis of College Reviews. *International Journal Of Engineering Development And Research, 5*(2), 319-322.

Castella, Q., & Sutton, C. (2014). Word Storm: Multiples of Word Clouds for Visual Comparison of Documents. *WWW*, 665-675.

Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, K. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research, 16*, 321-357.

Chung, J. Y., & Buhalis, D. (2008). Web 2.0: A Study of Online Travel Community. *Information and Communication Technologies in Tourism*, 70-81.

Deng, N., Tian, Y., & Zhang, C. (2012). *Support Vector Machines: Optimization Based Theory, Algorithms, and Extensions.* USA: Chapman and Hall/CRC.

Dragut, E., Fang, F., Sistla, P., & Yu, C. (2009). *Stop Word and Related Problems in Web Interface Integration.* Chicago: University of Illinois.

El-Khair, I. A. (2009). Term Weighting. *Encyclopedia of Database Systems, 1*, 3037-3040.

Feldman, R., & Sanger, J. (2007). *The Text Mining Handbook "Advanced Approaches in Analyzing Unstructured Data".* Cambridge: University Press.

Fletcher, R. (1987). *Practical Methods of Optimization.* USA: John Wiley & Sons.

Garreta, R., & Moncecchi, G. (2013). *Learning Scikit-Learn: Machine Learning in Python.* Berlin Heidelberg: Packt.

Gokgoz, E., & Subasi, A. (2015). Comparison of Decision Tree Algorithms for EMG Signal Classification using DWT. *Biomedical Signal Processing and Control, 18*, 138-144.

Han, J., Kamber, M., & Pei, J. (2011). *Data Mining: Concepts and Techniques.* USA: Morgan Kaufmann.

Hardle, W. K., Prastyo, D. D., & Hafner, C. M. (2014). Support Vector Machines with Evolutionary Model Selection for Default Prediction. Dalam *The Oxford Handbook of Applied Nonparametric and Semiparametric Econometrics and Statistics* (hal. 346-373). New York: Oxford University Press.

Hsu, C. W., Chang, C. C., & Lin, C. J. (2003). A Practical Guide to Support Vector Classification, 1-16.

Hu, M., & Liu, B. (2004, May 15). Dipetik February 12, 2019, dari https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

Huang, C. M., Lee, Y. J., Lin, D. K., & Huang, S. Y. (2007). Model Selection for Support Vector Machines Via Uniform Design. *Computational Statistics & Data Analysis, 52*(1), 335-346.

Hung, J., & Zhang, K. (2012). Examining Mobile Learning Trends 2003-2008: A Categorical Meta-Trend Analysis Using Text Mining Techniques. *Journal of Computing in Higher Education, 24*(1), 1-17.

Ismayanti. (2010). *Pengantar Pariwisata.* Jakarta: PT. Gramedia Widiasarana Indonesia.

Johnson, F., & Gupta, S. K. (2012). Web Content Mining Techniques: A Survey. *International Journal of Computer Applications, 47*(11), 44-50.

Kecman, V. (2005). Support Vector Machines – An Introduction. *StudFuzz, 177*, 1-47.

Koehn, P. (2009). *Statistical Machine Translation.* Cambridge: Cambridge University.

Kuhn, M., & Johnson, K. (2013). *Applied Predictive Modeling.* New York: Springer.

Kurniasari, S. R. (2018). *Implementasi SVM dan Asosiasi untuk Sentiment Analysis Data Ulasan The Phoenix Hotel Yogyakarta pada Situs TripAdvisor.* Yogyakarta: Universitas Islam Indonesia.

Kurniawan, T. (2017). *Implementasi Text Mining pada Analisis Sentimen Pengguna Twitter terhadap Media Mainstream Menggunakan Naïve Bayes Classifier dan Support Vector Machine.* Surabaya: Institut Teknologi Sepuluh Nopember.

Liu, B. (2010). "Sentiment analysis and subjectivity". Dalam Taylor, & Francis, *Handbook of Natural Language Processing, 2nd Edition.* USA: Boca Raton.

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies, 5*(1), 1-167.

Ma, C. M., Yang, W. S., & Cheng, B. W. (2014). How the Parameters of K-nearest Neighbor Algorithm Impact on the Best Classification Accucary: In Case of Parkinson Dataset. *Journal of Applied Sciences, 14*(2), 171-176.

Ma, H., Huang, W., Jing, Y., Yang, C., Han, L., Dong, Y., Ye, H., Shi, Y., Zheng, Q., Liu, L., & Ruan, C. (2019). Integrating Growth and Environmental Parameters to Discriminate Powdery Mildew and Aphid of Winter Wheat Using Bi-Temporal Landsat-8 Imagery. *Remote Sensing, 11*(846), 1-23.

Manning, C. D., Raghavan, P., & Schutze, H. (2008). *Introduction to Information Retrieval*. USA: Cambridge University Press.

Marafino, B. J., Davies, J. M., Bardach, N. S., Dean, M. L., & Dudley, R. A. (2014). N-gram Support Vector Machines for Scalable Procedure and Diagnosis Classification, with Applications to Clinical Free Text Data from the Intensive Care Unit. *Journal of the American Medical Informatics Association, 21*(5), 871-875.

Mohammad, S. M., Kiritchenko, S., & Zhu, X. (2013). NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets. *Proceedings of the Seventh International Workshop on Semantic Evaluation Exercises (SemEval-2013), 2*, 321-327.

Mukherjee, S. (2018). Sentiment Analysis of Reviews. *Encyclopedia of Social Network Analysis and Mining. 2nd Ed*., 2425-2434

Mulajati, M., & Hakim, R. B. (2017). Sentiment Analysis on Online Reviews Using Naive Bayes Classifier (Case Study : Garuda Indonesia Airlines Passengers Reviews on Tripadvisor Site). *Indian J.Sci.Res, 17*(1), 274-280.

Pitana, I. G., & Gayatri. (2005). *Sosiologi Pariwisata*. Yogyakarta: Andi Offset.

Praptiwi, D. Y. (2018). *Analisis Sentimen Online Review Pengguna E-Commerce Menggunakan Metode Support Vector Machine dan Maximum Entropy*. Yogyakarta: Universitas Islam Indonesia.

Rahman, F. (2018). *Klasifikasi Emosi untuk Teks Berbahasa Indonesia pada Pengguna Twitter Mengenai Presiden Joko Widodo.* Surabaya: Institut Teknologi Sepuluh Nopember.

Rani, T. J., Anuradha, K., & Reddy, P. V. (2016). Sentiment Classification on Twitter Data Using Word N Gram Model. *International Journal of Technology and Engineering Science, 4*(1), 7032-7035.

Ratman, D. R. (2016). *Pembangunan Destinasi Pariwisata Prioritas 2016-2019.* Jakarta: Kementerian Pariwisata.

Reyhana, Z., Fithriasari, K., Atok, M., & Iriawan, N. (2018). Linking Twitter Sentiment Knowledge with Infrastructure Development. *Matematika, 34*, 91-102.

Rifqi, N., Maharani, W., & Shaufiah. (2011). Analisis dan Implementasi Klasifikasi Data Mining Menggunakan Jaringan Syaraf Tiruan dan Evolution Strategis. *Konferensi Nasional Sistem dan Informatika, 29*, 183-191.

Sain, H., & Purnami, S. W. (2015). Combine Sampling Support Vector Machine for Imbalanced Data Classification. *Procedia Computer Science, 72*, 59-66.

Salton, G., & Buckley, C. (1988). Term-Weighting Approaches in Automatic Text Retrieval. *Information Processing & Management, 24*(5), 513-523.

Santos, M. S., Soares, J. P., Abreu, P. H., Araujo, H., & Santos, J. (2018). Cross-Validation for Imbalanced Datasets: Avoiding Overoptimistic and Overfitting Approaches. *IEEE Computational Intelligence Magazine, 13*(4), 59-76.

Septiana, N., Ridok, A., & Dewi, C. (2013). Pengelompokan Dokumen Berita Berbahasa Indonesia Menggunakan Fuzzy C-Means. *Respository Jurnal Mahasiswa PTIIK UB, 1*(7).

Shaheen, R., Ahsan, A., & Anwar, Z. (2018). Requirements Management For Market Driven Software Products — Key Issues. *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, 1-6.

Sokolova, M., & Lapalme, G. (2009). A Systematic Analysis of Performance Measures for Classification Tasks. *Information Processing and Management, 45*(4), 427-437.

Sun, Y., Wong, A. K., & Kamel, M. S. (2009). Classification of Imbalanced Data: A Review. *International Journal of Pattern Recognition and Artificial Intelligence, 23*(4), 687-719.

Syaani, F. A. (2019). *Pengelompokan Incident Dalam Lingkungan Kerja Warehouse Menggunakan Text Mining.* Surabaya: Institut Teknologi Sepuluh Nopember.

Toman, M., Tesar, R., & Jezek, K. (2006). Influence of Word Normalization on Text Classification. *In Proceedings of the 1st International Conference on Multidisciplinary Information Sciences & Technologies, 2*, 354-358.

TripAdvisor. (2017). Dipetik Februari 13, 2019, dari TripAdvisor: https://tripadvisor.mediaroom.com/us-about-us

Tripathy, A., Agrawal, A., & Rath, S. K. (2016). Classification of Sentiment Reviews using N-gram Machine Learning Approach. *Expert Systems With Applications, 57*, 117–126.

Turland, M. (2010). *Php|Architect's Guide to Web Scraping.* Canada: Marco Tabini & Associates, Inc.

Vapnik, V. N., & Chervonenkis, A. J. (1974). *Theory of Pattern Recognition.*

Wang, J., Neskovic, P., & Cooper, L. N. (2007). Improving Nearest Neighbor Rule with a Simple Adaptive Distance Measure. *Pattern Recognition Letters, 28*(2), 207-213.

Webb, A. R., & Copsey, K. D. (2011). *Statistical Pattern Recognition.* UK: Wiley.

Weiss, S. M. (2010). *Text Mining: Predictive Methods for Analyzing.* New York: Springer.

Zhang, S., Li, X., Zong, M., Zhu, X., & Wang, R. (2018). Efficient kNN Classification With Different Numbers of Nearest

Neighbors. *IEEE Transactions on Neural Networks and Learning Systems, 29*(5), 1774-1785.

Zheng, W., Wang, H., Ma, L., & Wang, R. (2015). An Improved k-Nearest Neighbor Classifcation Algorithm Using Shared Nearest Neighbor Similarity. *Metallurgical & Mining Industry*, (10), 133-137.

(Halaman ini sengaja dikosongkan)

LAMPIRAN

Lampiran 1. *Syntax Web Scraping* Menggunakan *Rstudio*

```
library(rvest)
for(i in 1:389)
{
  url <- paste0("https://www.tripadvisor.com/Attraction_Review-g790291-
d320054-Reviews-or",i,"0-Borobudur_Temple-
Borobudur_Magelang_Central_Java_Java.html")

  reviews <- url %>%
    read_html() %>%
    html_nodes("#REVIEWS .review-container")

  id <- reviews %>%
    html_node(".quote a") %>%
    html_attr("id")

  quote <- reviews %>%
    html_node(".quote span") %>%
    html_text()

  review <- reviews %>%
    html_node(".entry .partial_entry") %>%
    html_text()
  reviewnospace <- gsub("\n", "", review)
  df<-data.frame(id,quote,reviewnospace, stringsAsFactors = FALSE)
  #data.frame(id, quote, review, stringsAsFactors = FALSE) %>% View()
  write.table(df,file="D:/inggrisbor.csv",append=TRUE,row.names =
FALSE)
}
```

Lampiran 2. Data Ulasan
Sumber: *www.tripadvisor.com*, 11 Maret 2019

No	Ulasan
1	A UNESCO world heritage site, this place is well preserved and so clean. Excellent toilet facilities. A combine ticket for Borobudur and Prambanan temples (valid for 2 consecutive days) makes a cheaper option. Take a guide at prescribed rates. Our guide spoke very good English and was very knowledgeable. One has to climb nine levels (51metres) but the climb is not tiring as you get to rest at each level to admire the carvings and get to know the significance of each level. This is where the guide comes in handy. Once you are at the top admire the all around view. Once you come down buy some local merchandise and help the local economy.
2	It is a very large Buddhist temple, you need to be early around 8:00 when there is not so much tourists. The visit took me around hour and half without guide. If you have guide, it can be a lot more. I took combined ticket with Prambanan temple is it about hour away with scooter. You can do both in one day.
3	this tourist place is very good, we can see the view of the city of Magelang from the top of the temple. besides that this temple also has a hindu -buddha style architecture which is indeed very beautiful. no doubt many foreign and domestic tourists crowded this tourist spot
⋮.	⋮
902	Borobudur was an amazing historical place to visit. The view from above the temple was magnificent. My kids enjoyed climbing up the temple. It's best to visit the place in the morning when it wasn't that
⋮.	⋮
3591	The history and the architecture at this site is absolutely amazing! I'll always remember the day we spent there as an incredible journey back in time.

Lampiran 3. *Syntax Text Preprocessing* Menggunakan *Python*

```
#Import package
import string
import re
import nltk
import csv
import numpy as np
import pandas as pd
from nltk.tokenize import sent_tokenize, word_tokenize
from string import digits
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import CountVectorizer,
TfidfVectorizer
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.metrics import roc_curve, auc, confusion_matrix, f1_score,
precision_score, recall_score, accuracy_score, classification_report
from sklearn.model_selection import KFold
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn import model_selection
import imblearn
from imblearn.over_sampling import SMOTE
from sklearn import metrics

#Import data
df = pd.read_csv('D:/datafix.csv',encoding='cp1252',sep=";")
df.head()

review=df['Text']

#Case folding
review = review.apply(lambda x: " ".join(x.lower() for x in x.split()))
review

#Menghapus tanda baca
review = review.str.replace('[^\w\s]','')
review
```

Lampiran 3. *Syntax Text Preprocessing* Menggunakan *Python* (lanjutan)

```
#Menghapus angka
review = review.str.replace("\d"," ")
review

#Menghapus spasi berlebih
review = review.str.replace('\s+', ' ')
review

#Sinonim dan typo
kata = {"abt":"about","aug ":"august ","angkorwat":"angkor wat","atleast
":"at least","bas relief":"basrelief"," cos ":" because ","beautifull ":"
beautiful ","budda ":"buddha ","chandi ":"temple ","feb ":"february ","bro
":"brother ","jan ":"january ","yk ":"yogyakarta ","jogja ":" yogyakarta
","yogya ":"yogyakarta ","rout ":"route ","jul ":"july ","km ":"kilometer
","kms":"kilometer","lil ":"little ","man made":"manmade","makesure
":"make sure","max ":"maximum ","metre ":"meter ","mustvisitnow":"
must visit now","pic ":"picture ","pics ":"picture ","pict ":"picture","ppl
":"people ","earth quake":"earthquake","resto ":"restaurant ","sept ":
"september ","sep ":"september ","opp ":"opposite "," on site":"onsite","opt
":"optative ","stonefretworkwall":"stone fretwork wall","mtmerapi":
"mount merapi","non violence":"nonviolence", "sign board":"signboard","t
shirt":"tshirt","over priced":"overpriced","zig zag":"zigzag","phototaking":
"photo taking","softgloomy":"soft gloomy","smal ":"small ","faint hearted"
:"fainthearted","dontt":"dont","myst ":"mystic ","mustsee":"must see",
"mustvisit":"must visit","sun glass ":"sunglass ","sun glasses":"sunglass",
"giftshop":"gift shop","jun ":"june ","blackstones":"black stones","motor
bike":"motorbike","pickup":"pick up","trans jogja":"transjogja","presunrise
":"pre sunrise","stupaas":"stupa ","knownso":"known so","ligh ":"light ",
"lesscrowded":"less crowded","lightstick":"light stick","legcramping":"leg
cramping","key points":"keypoint","key point ":"keypoint ","bellshaped":
"bell shaped","infact":"in fact","inorder":"in order","instal ":"install ",
"unescos":"unesco","view point":"viewpoint","bath room":"bathroom",
"vesaks":"vesak","wellmaintained":"well maintained","wellpreserved":
"well preserved","well managed":"wellmanaged","welldeserving":"well
deserving","pricetag":"price tag","pricegouging":"price gouging",
"keychains":"keychain","looksoverall":"looks overall","eventhough":
"even though","equiv ":"equivalent ","wonderfull ":"wonderful ","rupia ":
```

Lampiran 3. *Syntax Text Preprocessing* Menggunakan *Python* (lanjutan)

```
"rupiah ","over crowded":"overcrowded","pre dawn":"predawn","tourguide
":"tour guide ","well known":"wellknown","world famous":"worldfamous"
,"mini train":"minitrain","mind blowing":"mindblowing","nonindonesian":
"non indonesian","nonlocals":"non locals","midmorming":"mid morning",
"mid week":"midweek","midmorning":"mid morning","wheel chair":
"wheelchair","photoshoot ":"photo shoot ","trip advisor":"tripadvisor",
"eventhou ":"even though ","aircondition ":"airconditioned","breaktaking":
"breathtaking","vist ":"visit ","mobbedwell":"mobbed well","worldclass":
"world class","worth while":"worthwhile","clothesthere":"cloth there",
"cigaret ":"cigarette ","bucketlist":"bucket list","doesns":"doesnt","coffe ":
"coffee ","crow ":"crowd ","umbrelles":"umbrella","motor cycle":
"motorbike","oct ":"october ","ornamen ":"ornament ","praise worthy":
"praiseworthy","pre dawn":"predawn","prambana ":"prambanan ","indon ":
"indonesian ","imho":"in my humble opinion","imo ":"in my opinion ","mt
":"mount ","nowaday ":"nowadays ","donts":"dont","wash room ":
"washroom ","guide book":"guidebook","scupltures":"sculpture",
"footwearrubber":"footwear rubber","foot path":"footpath","flash light":
"flashlight","torch ":"flashlight ","torches":"flashlight","torchlight":
"flashlight"," oke":" ok", "wc ":"toilet ","guid ":"guide ","non sense":
"nonsense","sun block":"sunblock","aweinspiring":"awe inspiring","car
park":"carpark","mind boggling":"mindboggling","south east asia":
"southeast asia","paparazzis":"paparazzi","texting":"text"}

def replace_all(text, dic):
        for i, j in dic.items():
            text = text.replace(i, j)
        return text

import collections
from collections import OrderedDict
dic = OrderedDict(kata)

datachange = []
for line in review:
    result = replace_all(line, dic)
    datachange.append(result)
datachange
```

Lampiran 3. *Syntax Text Preprocessing* Menggunakan *Python* (lanjutan)

```
review=pd.DataFrame(datachange,columns=["bersih"])
review.head()

review1=review['bersih']

#Lemmatization1
lemma = WordNetLemmatizer()
review1 = review1.apply(lambda x: " ".join([lemma.lemmatize(word,
pos="v") for word in x.split()]))
review1

#Lemmatization2
lemma = WordNetLemmatizer()
review1 = review1.apply(lambda x: " ".join([lemma.lemmatize(word) for
word in x.split()]))
review1

#Stopword
stopword = open('D:/stopwords_en.txt','r').read()
review1 = review1.apply(lambda x: " ".join(x for x in x.split() if x not in
stopword))
review1

#Menyimpan data hasil cleaning
cleaning=pd.DataFrame(review1)
cleaning.to_csv(r'D:/clean fix 1.csv')

#Tokenizing
token=[]
for line in review1 :
    b = nltk.word_tokenize(line)
    token.append(b)
token_df=pd.DataFrame(token)
token_df
```

Lampiran 4. Hasil *Text Preprocessing*

No	Ulasan
1	unesco world heritage site well preserve clean excellent toilet facility combine ticket prambanan valid consecutive cheaper option guide prescribe guide speak good english knowledgeable climb level meter climb rest level admire carve significance level guide handy admire view buy local merchandise local economy
2	large tourist visit hour half guide guide combine ticket prambanan hour scooter
3	tourist good view city magelang hindu style architecture beautiful doubt foreign domestic tourist crowd tourist spot
⋮	⋮
902	amaze historical visit view magnificent enjoy climb visit morning
903	largest world recommendation love culture walk amaze blend cultural cultural heritage streak relief amaze love
904	sunrise life magical wait dark sunrise rupiah snack coffee include finish picture lapse watch
⋮	⋮
3590	hour beautiful scenic road rice field small amaze sculpture show story visit prambanan show night javanese style dinner buffet century find untouchable huge mountain stair lead level stone carve sculpture small statue carve story biggest javanese kingdom touch statue circle counter clockwise statue grant good luck well amaze biggest overlook java surround amaze view breathtaking comeback highly recommend amaze love world happy
3591	history architecture site absolutely amaze remember spend incredible journey

Lampiran 5. *Syntax Labelling Lexicon* Menggunakan *RStudio*

```
library(tm)
kalimat2<-read.csv("D:/clean_3.csv",header=TRUE)
#Scoring
positif <- scan("D:/positive-
words.txt",what="character",comment.char=";")
negatif <- scan("D:/negative-
words.txt",what="character",comment.char=";")
score.sentiment = function(kalimat2, positif, negatif,.progress='none')
{
  require(plyr)
  require(stringr)
  scores = laply(kalimat2, function(kalimat, positif, negatif)
  {
    kalimat = gsub('[[:punct:]]', '', kalimat)
    kalimat = gsub('[[:cntrl:]]', '', kalimat)
    kalimat = gsub('\\d+', '', kalimat)
    kalimat = tolower(kalimat)
    list.kata = str_split(kalimat, '\\s+')
    kata2 = unlist(list.kata)
    positif.matches = match(kata2, positif)
    negatif.matches = match(kata2, negatif)
    positif.matches = !is.na(positif.matches)
    negatif.matches = !is.na(negatif.matches)
    score = sum(positif.matches) - (1*sum(negatif.matches))
return(score)
  }, positif, negatif, .progress=.progress )
  scores.df = data.frame(score=scores, text=kalimat2)
  return(scores.df)
}
hasil = score.sentiment(kalimat2$text, positif, negatif)
View(hasil)

#CONVERT SCORE TO SENTIMENT
hasil$klasifikasi<- ifelse(hasil$score<0, "Negatif","Positif")
hasil$klasifikasi
View(hasil)
#EXCHANGE ROW SEQUENCE
data <- hasil[c(3,1,2)]
View(data)
write.csv(data, file = "D:/hasillabel.csv")
```

Lampiran 6. Hasil *Labelling Lexicon*

No	sentimen	label	score	Text
1	0	Positif	9	unesco world heritage site well preserve clean excellent toilet facility combine ticket prambanan valid consecutive cheaper option guide prescribe guide speak good english knowledgeable climb level meter climb rest level admire carve significance level guide handy admire view buy local merchandise local economy
2	0	Positif	0	large tourist visit hour half guide guide combine ticket prambanan hour scooter
⋮	⋮	⋮	⋮	⋮
1639	1	Negatif	-3	spiritual site lose commercialism government charge highprice tourist visit site inside direction tout inside sell tourist rubbish official student card child ticket card exist spain force walk market sell junk appreciate live ruin spiritual site
1640	0	Positif	4	visit amaze security staff pleasant helpful pay extra sunrise sunset pay standard ticket stick load tourist good picture pay extra sunrise sunset crowd amaze photo
⋮	⋮	⋮	⋮	⋮
3591	0	Positif	2	history architecture site absolutely amaze remember spend incredible journey

Lampiran 7. *Syntax* Karakterisik Ulasan dengan *Python*

```
#5 kata dengan frekuensi terbanyak (unigram)
vec = CountVectorizer()
vec.fit(textclean)
tf=np.sum(vec.transform(textclean),axis=0)
tf1 = np.squeeze(np.asarray(tf))
termfreq_df =
pd.DataFrame([tf1],columns=vec.get_feature_names()).transpose()
termfreq_df.head()

termfreq_df.columns = ['Frekuensi']
termfreq_df.sort_values(by='Frekuensi', ascending=False).iloc[:5]

#5 kata dengan frekuensi terbanyak (bigram)
vec = CountVectorizer(ngram_range=(2,2))
vec.fit(textclean)
tf=np.sum(vec.transform(textclean),axis=0)
tf1 = np.squeeze(np.asarray(tf))
termfreq_df =
pd.DataFrame([tf1],columns=vec.get_feature_names()).transpose()
termfreq_df.head()

termfreq_df.columns = ['Frekuensi']
termfreq_df.sort_values(by='Frekuensi', ascending=False).iloc[:5]

#5 kata dengan frekuensi terbanyak (trigram)
vec = CountVectorizer(ngram_range=(3,3))
vec.fit(textclean)
tf=np.sum(vec.transform(textclean),axis=0)
tf1 = np.squeeze(np.asarray(tf))
termfreq_df =
pd.DataFrame([tf1],columns=vec.get_feature_names()).transpose()
termfreq_df.head()

termfreq_df.columns = ['Frekuensi']
termfreq_df.sort_values(by='Frekuensi', ascending=False).iloc[:5]
```

Lampiran 7. *Syntax* Karakterisik Data dengan *Python* (lanjutan)

```
#Pie chart sentimen
label=datanew.groupby('label').size().reset_index(name='counts')
fig,
ax=plt.subplots(figsize=(12,7),subplot_kw=dict(aspect="equal"),dpi=300)
datacount=label['counts']
kategori=label['label']
def func(pct, allvals):
   absolute=int(pct/100.*np.sum(allvals))
   return "{:.2f}% ({:d})".format(pct, absolute)
wedges, texts, autotexts=ax.pie(datacount,
                        autopct=lambda pct: func(pct, datacount),
                        textprops=dict(color="w"),
                        colors=plt.cm.Set1.colors,
                        startangle=140)
ax.legend(wedges, kategori, title="Sentimen", loc="center left",
bbox_to_anchor=(1,0,0.5,1))
plt.setp(autotexts, size=10, weight=700)
plt.savefig('pie chart',dpi=300)
plt.show()
```

Lampiran 8. *Syntax Term Weighting Unigram* Menggunakan *Python*

```
#Import data hasil labelling
datanew = pd.read_csv('D:/label fix 2.csv',encoding='cp1252',sep=";")
datanew.head()
textclean=datanew['text']
sentimen=datanew['sentimen']
#Term frequency
vec = CountVectorizer(min_df=20).fit(textclean)
X = vec.fit_transform(textclean)
df_1 = pd.DataFrame(X.toarray(), columns=vec.get_feature_names())
df_df=pd.concat([datanew['sentimen'],df_1], axis=1)
df_df.tail()
#TF-IDF
vec1 = TfidfVectorizer(min_df=20)
tfidf = vec1.fit_transform(textclean)
tfidf1 = pd.DataFrame(tfidf.toarray(),
                columns=vec1.get_feature_names())
tfidf1_=pd.concat([datanew['sentimen'],tfidf1], axis=1)
tfidf1_.head()
```

Lampiran 9. Nilai TF-IDF *Unigram*

		Ulasan							
		1	2	...	1000	1001	...	3590	3591
Sentimen		0	0	...	0	0	...	0	0
Kata	***absolute***	0	0	...	0	0	...	0	0
	absolutely	0	0	...	0	0	...	0	0,4
	accept	0	0	...	0	0	...	0	0
	access	0	0	...	0	0	...	0	0
	active	0	0	...	0	0	...	0	0
	activity	0	0	...	0	0	...	0	0
	add	0	0	...	0	0	...	0	0
	additional	0	0	...	0	0	...	0	0
	⋮	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮
	garden	0	0	...	0	0	...	0	0
	gate	0	0	...	0	0	...	0	0
	general	0	0	...	0	0	...	0	0
	giant	0	0	...	0	0	...	0	0
	gift	0	0	...	0	0	...	0	0
	glad	0	0	...	0	0	...	0	0
	good	0,1	0	...	0,2	0	...	0,1	0
	⋮	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮
	open	0	0	...	0	0	...	0	0
	opinion	0	0	...	0	0	...	0	0
	opportunity	0	0	...	0	0	...	0	0
	opt	0	0	...	0	0	...	0	0
	option	0,2	0	...	0	0	...	0	0
	order	0	0	...	0	0	...	0	0
	⋮	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮
	worthwhile	0	0	...	0	0	...	0	0
	worthy	0	0	...	0	0	...	0	0
	wow	0	0	...	0	0	...	0	0
	write	0	0	...	0	0	...	0	0
	wrong	0	0	...	0	0	...	0	0

Lampiran 10. Nilai TF-IDF *Bigram*

Ulasan	Kata (*Feature*)						
	amaze architecture	...	***heritage site***	...	***site well***	...	***worth visit***
1	0	...	0,3	...	0,4	...	0
2	0	...	0	...	0	...	0
3	0	...	0	...	0	...	0
4	0	...	0	...	0	...	0
5	0	...	0	...	0	...	0
6	0	...	0	...	0	...	0
7	0	...	0	...	0	...	0
⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
1076	0	...	0	...	0	...	0,3
1077	0	...	0	...	0	...	0,4
1078	0	...	0	...	0	...	0
1079	0	...	0	...	0	...	0
1080	0	...	0	...	0	...	0
1081	0	...	0	...	0	...	0
1082	0	...	0	...	0	...	0
⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
3585	0	...	0	...	0	...	0
3586	0	...	0	...	0	...	0
3587	0	...	0	...	0	...	0
3588	0	...	0	...	0	...	0
3589	0	...	0	...	0	...	0
3590	0	...	0	...	0	...	0
3591	0	...	0	...	0	...	0

Lampiran 11. Nilai TF-IDF *Trigram*

Ulasan	**Kata (*Feature*)**				
	absolutely worth visit	**...**	***site well preserve***	**...**	***worth visit visit***
1	0	...	0,5	...	0,4
2	0	...	0	...	0
3	0	...	0	...	0
4	0	...	0	...	0
5	0	...	0	...	0
6	0	...	0	...	0
7	0	...	0	...	0
⋮	⋮	⋱	⋮	⋱	⋮
1076	0	...	0	...	0
1077	0	...	0	...	0
1078	0	...	0	...	0
1079	0	...	0	...	0
1080	0	...	0	...	0
1081	0	...	0	...	0
1082	0	...	0	...	0
⋮	⋮	⋱	⋮	⋱	⋮
3585	0	...	0	...	0
3586	0	...	0	...	0
3587	0	...	0	...	0
3588	0	...	0	...	0
3589	0	...	0	...	0
3590	0	...	0	...	0
3591	0	...	0	...	0

Lampiran 12. *Syntax* Klasifikasi dengan *Python*

```
tfidf_new=tfidf1.values
tfidf_new

Y=pd.DataFrame(sentimen)
Y1=pd.DataFrame.as_matrix(Y)
Y1

Y1_1=np.ravel(Y1)
Y1_1

#Mendefinisikan variabel X dan Y
X_new, Y_new=tfidf_new, Y1_1

#Menentukan k terbaik untuk KNN
neighbors=np.arange(1,50)
n=10
kfold=KFold(n_splits=n, random_state = 200, shuffle=True)
smote=SMOTE(random_state=123)
total=dict()
auc=dict()
cm=dict()
for i,k in enumerate(neighbors):
    cm[i]=[]
    total[i]=[]
    auc[i]=[]
    knn=KNeighborsClassifier(n_neighbors=k)
    for train, test in kfold.split(X_new, Y_new):
        (X_new[train], X_new[test])
        (Y_new[train], Y_new[test])
        X_trainsmote,
Y_trainsmote=smote.fit_sample(X_new[train],Y_new[train])
        knn.fit(X_trainsmote,Y_trainsmote)
        prediksi = knn.predict(X_new[test])
        cm[i].append((confusion_matrix(Y_new[test],prediksi)).astype(float))
for i in range(len(neighbors)):
    total[i]=[]
    auc[i]=[]
    for j in range(n):
```

Lampiran 12. *Syntax* Klasifikasi dengan *Python* (lanjutan)

```
        total[i].append(sum(sum(cm[i][j])))
auc[i].append(0.5*((cm[i][j][1,1]/(cm[i][j][1,0]+cm[i][j][1,1]))+(cm[i][j][0,
0]/(cm[i][j][0,0]+cm[i][j][0,1]))))
auc1=np.empty(len(neighbors))
for i in range(len(neighbors)):
    auc1[i]=np.mean(auc[i])
plt.plot(neighbors, auc1, label='AUC', marker='o', markerfacecolor='red',
markersize=5)
plt.legend()
plt.xlabel('Nilai k')
plt.ylabel('AUC')
plt.savefig('K uni label 2',dpi=300)
plt.show()

print(max(auc1))

#Klasifikasi KNN dengan k terbaik
smote=SMOTE(random_state=123)
kfold = KFold(n_splits=10, random_state = 200, shuffle=True)
for train, test in kfold.split(X_new, Y_new):
    (X_new[train], X_new[test])
    (Y_new[train], Y_new[test])
    X_trainsmote,
Y_trainsmote=smote.fit_sample(X_new[train],Y_new[train])
    knn = KNeighborsClassifier(n_neighbors=2).fit(X_trainsmote,
Y_trainsmote)
    prediksi = knn.predict(X_new[test])
    print("\nConfusion Matrix: \n", metrics.confusion_matrix(Y_new[test],
prediksi))
    fpr,tpr,_=metrics.roc_curve(Y_new[test],prediksi)
    auc_=metrics.auc(fpr,tpr)
    print ("AUC = {:.4f}".format(auc_))
    print("Akurasi Prediksi = ", accuracy_score(Y_new[test], prediksi))
    print()
    label = ['Positif', 'Negatif']
    print(metrics.classification_report(Y_new[test], prediksi,
target_names=label))
    print("-------------------------------------------------------------------------------
-----------------------")
```

Lampiran 12. *Syntax* Klasifikasi dengan *Python* (lanjutan)

```
#Menentukan C terbaik untuk SVM kernel linear
crange = [0.01, 0.1, 1, 10, 100]
n=10
kfold=KFold(n_splits=n, random_state = 123, shuffle=True)
smote=SMOTE(random_state=123)
total=dict()
auc=dict()
cm=dict()
for i,c in enumerate(crange):
   cm[i]=[]
   total[i]=[]
   auc[i]=[]
   svm=SVC(kernel='linear', C=c)
   for train, test in kfold.split(X_new, Y_new):
      (X_new[train], X_new[test])
      (Y_new[train], Y_new[test])
      X_trainsmote,
Y_trainsmote=smote.fit_sample(X_new[train],Y_new[train])
      svm.fit(X_trainsmote,Y_trainsmote)
      prediksi = svm.predict(X_new[test])
      cm[i].append((confusion_matrix(Y_new[test],prediksi)).astype(float))
for i in range(len(crange)):
   total[i]=[]
   auc[i]=[]
   for j in range(n):
      total[i].append(sum(sum(cm[i][j])))
auc[i].append(0.5*((cm[i][j][1,1]/(cm[i][j][1,0]+cm[i][j][1,1]))+(cm[i][j][0,
0]/(cm[i][j][0,0]+cm[i][j][0,1]))))
auc1=np.empty(len(crange))
for i in range(len(crange)):
   auc1[i]=np.mean(auc[i])
print(auc1)
```

Lampiran 12. *Syntax* Klasifikasi dengan *Python* (lanjutan)

```
#Klasifikasi SVM kernel linear dengan C terbaik
smote=SMOTE(random_state=123)
kfold = KFold(n_splits=10, random_state = 123, shuffle=True)
for train, test in kfold.split(X_new, Y_new):
    (X_new[train], X_new[test])
    (Y_new[train], Y_new[test])
    X_trainsmote,
Y_trainsmote=smote.fit_sample(X_new[train],Y_new[train])
    sv = SVC(kernel='linear', C=1)
    svm = sv.fit(X_trainsmote, Y_trainsmote)
    prediksi = svm.predict(X_new[test])
    print("\nConfusion Matrix: \n", metrics.confusion_matrix(Y_new[test],
prediksi))
    fpr,tpr,_=metrics.roc_curve(Y_new[test],prediksi)
    auc_=metrics.auc(fpr,tpr)
    print ("AUC = {:.4f}".format(auc_))
    print("Akurasi Prediksi: ", accuracy_score(Y_new[test], prediksi))
    print()
    label = ['Positif', 'Negatif']
    print(metrics.classification_report(Y_new[test], prediksi,
target_names=label))
    print("-------------------------------------------------------------------------------
-----------------------")
```

Lampiran 12. *Syntax* Klasifikasi dengan *Python* (lanjutan)

```
#Menentukan gamma terbaik untuk SVM kernel RBF
grange = [2**-15, 2**-13, 2**-11, 2**-9, 2**-7, 2**-5, 2**-3, 2**-1,
2**1, 2**3]
n=10
kfold=KFold(n_splits=n, random_state = 123, shuffle=True)
smote=SMOTE(random_state=123)
total=dict()
auc=dict()
cm=dict()
for i,g in enumerate(grange):
    cm[i]=[]
    total[i]=[]
    auc[i]=[]
    svm=SVC(kernel='rbf', C=100, gamma=g)
    for train, test in kfold.split(X_new, Y_new):
        (X_new[train], X_new[test])
        (Y_new[train], Y_new[test])
        X_trainsmote,
Y_trainsmote=smote.fit_sample(X_new[train],Y_new[train])
        svm.fit(X_trainsmote,Y_trainsmote)
        prediksi = svm.predict(X_new[test])
        cm[i].append((confusion_matrix(Y_new[test],prediksi)).astype(float))
for i in range(len(grange)):
    total[i]=[]
    auc[i]=[]
    for j in range(n):
        total[i].append(sum(sum(cm[i][j])))
auc[i].append(0.5*((cm[i][j][1,1]/(cm[i][j][1,0]+cm[i][j][1,1]))+(cm[i][j][0,
0]/(cm[i][j][0,0]+cm[i][j][0,1]))))
auc1=np.empty(len(grange))
for i in range(len(grange)):
    auc1[i]=np.mean(auc[i])
print(auc1)
```

Lampiran 12. *Syntax* Klasifikasi dengan *Python* (lanjutan)

```
#Klasifikasi SVM kernel RBF dengan C dan gamma terbaik
smote=SMOTE(random_state=123)
kfold = KFold(n_splits=10, random_state = 123, shuffle=True)
for train, test in kfold.split(X_new, Y_new):
   (X_new[train], X_new[test])
   (Y_new[train], Y_new[test])
   X_trainsmote,
Y_trainsmote=smote.fit_sample(X_new[train],Y_new[train])
   sv = SVC(kernel='rbf', C=100, gamma=2**-9)
   svm = sv.fit(X_trainsmote, Y_trainsmote)
   prediksi = svm.predict(X_new[test])
   print("\nConfusion Matrix: \n", metrics.confusion_matrix(Y_new[test],
prediksi))
   fpr,tpr,_=metrics.roc_curve(Y_new[test],prediksi)
   auc_=metrics.auc(fpr,tpr)
   print ("AUC = {:.4f}".format(auc_))
   print("Akurasi Prediksi: ", accuracy_score(Y_new[test], prediksi))
   print()
   label = ['Positif', 'Negatif']
   print(metrics.classification_report(Y_new[test], prediksi,
target_names=label))
   print("-------------------------------------------------------------------------------
-----------------------")
```

Lampiran 13. *Confusion Matrix* Metode K-NN (*Unigram*)

Kelas Aktual	**Kelas Prediksi**		**Total**
	Positif	Negatif	
Positif	1056	2062	**3118**
Negatif	53	420	**473**
Total	**1109**	**2482**	**3591**

Lampiran 14. *Confusion Matrix* Metode K-NN (*Bigram*)

Kelas Aktual	**Kelas Prediksi**		**Total**
	Positif	Negatif	
Positif	1440	1678	**3118**
Negatif	173	300	**473**
Total	**1613**	**1978**	**3591**

Lampiran 15. *Confusion Matrix* Metode K-NN (*Trigram*)

Kelas Aktual	Kelas Prediksi		Total
	Positif	Negatif	
Positif	2913	205	**3118**
Negatif	420	53	**473**
Total	**3333**	**258**	**3591**

Lampiran 16. *Confusion Matrix* SVM *Kernel Linear* (*Unigram*)

Kelas Aktual	Kelas Prediksi		Total
	Positif	Negatif	
Positif	2773	345	**3118**
Negatif	115	358	**473**
Total	**2888**	**703**	**3591**

Lampiran 17. *Confusion Matrix* SVM *Kernel Linear* (*Bigram*)

Kelas Aktual	Kelas Prediksi		Total
	Positif	Negatif	
Positif	2578	540	**3118**
Negatif	291	182	**473**
Total	**2869**	**722**	**3591**

Lampiran 18. *Confusion Matrix* SVM *Kernel Linear* (*Trigram*)

Kelas Aktual	Kelas Prediksi		Total
	Positif	Negatif	
Positif	638	2480	**3118**
Negatif	65	408	**473**
Total	**704**	**2888**	**3591**

Lampiran 19. Rata-rata Nilai AUC berbagai Kombinasi Parameter C dan γ SVM Kernel RBF

Unigram		*C*				
		0,01	0,1	1	10	**100**
γ	2^{-15}	0,7579	0,7579	0,7579	0,7579	0,7579
	2^{-13}	0,7584	0,7584	0,7584	0,7584	0,7818
	2^{-11}	0,7579	0,7579	0,7579	0,7563	0,813
	2^{-9}	0,7579	0,7579	0,7579	0,796	**0,8234**
	2^{-7}	0,7583	0,7583	0,7608	0,8187	0,8002
	2^{-5}	0,76	0,76	0,7997	0,8089	0,7721
	2^{-3}	0,7494	0,754	0,7876	0,7545	0,7487
	2^{-1}	0,6233	0,6389	0,7132	0,7342	0,7342
	2^{1}	0,5	0,5	0,5181	0,5337	0,5337
	2^{3}	0,5	0,5	0,5021	0,5021	0,5021

Bigram		*C*				
		0,01	**0,1**	1	10	100
γ	2^{-15}	0,6017	0,6017	0,6017	0,6017	0,6017
	2^{-13}	0,6022	0,6022	0,6022	0,6022	0,589
	2^{-11}	0,605	0,605	0,605	0,605	0,6009
	2^{-9}	0,6052	0,6052	0,6052	0,6005	0,5822
	2^{-7}	0,6006	0,6006	0,5957	0,5937	0,5808
	2^{-5}	0,6068	**0,6068**	0,6006	0,5892	0,5845
	2^{-3}	0,595	0,5846	0,5858	0,5787	0,5573
	2^{-1}	0,505	0,5623	0,5781	0,5541	0,5459
	2^{1}	0,5015	0,5288	0,5311	0,5368	0,5376
	2^{3}	0,5015	0,5175	0,5157	0,515	0,5128

Lampiran 19. Rata-rata Nilai AUC berbagai Kombinasi Parameter *C* dan γ SVM *Kernel* RBF (lanjutan)

Trigram		*C*				
		0,01	0,1	1	**10**	100
γ	2^{-15}	0,5184	0,5184	0,5184	0,5184	0,5184
	2^{-13}	0,5185	0,5185	0,5185	0,5185	0,5185
	2^{-11}	0,5184	0,5184	0,5184	0,5184	0,5333
	2^{-9}	0,5184	0,5184	0,5184	0,5204	0,5394
	2^{-7}	0,5185	0,5185	0,5185	0,5293	0,5348
	2^{-5}	0,5152	0,5152	0,5174	**0,5417**	0,5294
	2^{-3}	0,5163	0,5163	0,5351	0,5288	0,5246
	2^{-1}	0,5111	0,5216	0,5321	0,5224	0,5284
	2^{1}	0,5084	0,5165	0,5242	0,5262	0,5241
	2^{3}	0,5084	0,5179	0,5225	0,5258	0,5249

Lampiran 20. *Confusion Matrix* SVM *Kernel* RBF (*Unigram*)

Kelas Aktual	**Kelas Prediksi**		**Total**
	Positif	Negatif	
Positif	2773	345	**3118**
Negatif	115	358	**473**
Total	**2888**	**703**	**3591**

Lampiran 21. *Confusion Matrix* SVM *Kernel* RBF (*Bigram*)

Kelas Aktual	**Kelas Prediksi**		**Total**
	Positif	Negatif	
Positif	2229	889	**3118**
Negatif	233	240	**473**
Total	**2462**	**1129**	**3591**

Lampiran 22. *Confusion Matrix* SVM *Kernel* RBF (Trigram)

Kelas Aktual	**Kelas Prediksi**		**Total**
	Positif	Negatif	
Positif	704	2414	**3118**
Negatif	67	406	**473**
Total	**771**	**2820**	**3591**

Lampiran 23. Model SVM *Kernel* RBF (*Unigram*)

```
Kernel used:
        RBF Kernel: K(x,y) = exp(-0.001953125*(x-y)^2)

Classifier for classes: 0, 1

BinarySMO

-   100     * < 0 0 0 0 0.234797 ... 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0> * X]
+   1.5593 * < 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0 0.043843 0 0 0 0 0> * X]
-   100     * < 0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0> * X]
                                        ⋮
-   100     * < 0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0> * X]
-   100     * < 0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0> * X]
+   100     * < 0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0> * X]
-    41.1822

Number of support vectors: 2471
```

Lampiran 24. *Syntax Word Cloud Unigram* menggunakan *Python*

```
positif=datanew[datanew['sentimen']==0]
positif=positif['text']
negatif=datanew[datanew['sentimen']==1]
negatif=negatif['text']
a=str(positif)
positif=re.sub(r"","",a)
b=str(negatif)
negatif=re.sub(r"","",b)

import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
%matplotlib inline
from subprocess import check_output
from wordcloud import WordCloud, STOPWORDS
mpl.rcParams['font.size']=12 #10
mpl.rcParams['savefig.dpi']=100 #72
mpl.rcParams['figure.subplot.bottom']=.1
from os import path
from PIL import Image
```

Lampiran 24. *Syntax Word Cloud Unigram* menggunakan *Python* (lanjutan)

```
#Wordcloud positif
wc = WordCloud(collocations = True,
                    background_color='white',
                    max_words=29,
                    max_font_size=200,
                    random_state=10,
                    width=800,
                    height=400,
                    colormap='viridis',
                    font_path='C:/Windows/Fonts/JosefinSans-Regular.ttf',
                   ).generate(positif)
print(wc)
plt.figure(figsize=(12,10))
plt.imshow(wc, interpolation="bilinear")
plt.axis("off")
plt.savefig('wordcloud positif',dpi=300)
plt.show()

#Wordcloud negatif
wc = WordCloud(collocations = True,
                    background_color='white',
                    max_words=25,
                    max_font_size=200,
                    random_state=10,
                    width=800,
                    height=400,
                    colormap='plasma',
                    font_path='C:/Windows/Fonts/JosefinSans-Regular.ttf',
                   ).generate(negatif)
print(wc)
plt.figure(figsize=(12,10))
plt.imshow(wc, interpolation="bilinear")
plt.axis("off")
plt.savefig('wordcloud negatif',dpi=300)
plt.show()
```

Lampiran 25. Surat Keterangan Pengambilan Data

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini, mahasiswa Departemen Statistika FMKSD ITS,

Nama : Rahayu Prihatini Saputri

NRP : 062115 4000 0040

menyatakan bahwa data yang digunakan dalam Tugas Akhir ini merupakan data sekunder yang diambil dari ~~penelitian / buku / Tugas Akhir / Thesis / Publikasi /~~ lainnya yaitu:

Sumber : *Website www.tripadvisor.com*

Keterangan : Data ulasan berbahasa Inggris wisatawan Candi Borobudur, dengan *keyword* "*borobudur temple*"

Surat pernyataan ini dibuat dengan sebenarnya. Apabila terdapat pemalsuan data maka saya siap menerima sanksi sesuai aturan yang berlaku.

Surabaya, Mei 2019

Rahayu Prihatini Saputri
NRP. 062115 4000 0040

Mengetahui,

Pembimbing Tugas Akhir I	Pembimbing Tugas Akhir II
Dra. Wiwiek Setya Winahju, M.S.	Dr. Dra. Kartika Fithriasari, M.Si.
NIP. 19560424 198303 2 001	NIP. 19691212 199303 2 002

BIODATA PENULIS

Penulis yang akrab disapa Rahayu lahir di Pasuruan, 3 Juni 1998, dari pasangan Bapak Henry Artiono dan Ibu Ika Cahyawati. Penulis telah menempuh pendidikan formal di SDN Kebonsari, SMPN 2 Pasuruan, dan SMAN 1 Pasuruan. Kemudian di tahun 2015 penulis diterima sebagai mahasiswa Departemen Statistika ITS. Selama masa perkuliahan, penulis aktif di divisi *Statistics Computer Course* (SCC) Himpunan Mahasiswa Statistika ITS (HIMASTA-ITS) sebagai staff *public relation* periode 2016-2017 dan manajer *public relation* pada periode 2017-2018. Selain itu penulis juga menjadi panitia di beberapa *event*, yakni Pekan Raya Statistika (PRS), Pekan Seni Mahasiswa ITS (PEKSIMITS), dll. Bagi pembaca yang ingin berdiskusi, memberikan saran, dan kritik mengenai Tugas Akhir ini, dapat menyampaikannya melalui *e-mail* rprihatini2@gmail.com.