PEMROSESAN BAHASA ALAMI

MENGGUNAKAN VECTOR SPACE

MEREDITA SUSANTY

Penerbit Universitas Pertamina

Pemrosesan Bahasa Alami – Menggunakan *Vector Space*

Penulis: Meredita Susanty

ISBN:

Editor: Uha Julaeha

Penerbit: Universitas Pertamina

Cetakan Pertama, 2023
Edisi Pertama, 2023

Jl Teuku Nyak Arief Simprug
Kebayoran Lama, Jakarta Selatan 12120
Telepon : 021-29044308
Website : https://universitaspertamina.ac.id/
Email : info@universitaspertamina.ac.id

Cetakan Pertama, 2023
Edisi Pertama, 2023

Susanty, Meredita

Pemrosesan Bahasa Alami – Menggunakan *Vector Space*

—Jakarta: Penerbit Universitas Pertamina,2023
1 jil., 150 hlm., 17,6 x 25 cm

ISBN. 978-623-94030-7-2

1. Ilmu Komputer
2. Kecerdasan Buatan
I. Judul
II. Susanty, Meredita

Tentang Penulis

Meredita Susanty adalah dosen tetap program studi Ilmu Komputer Universitas Pertamina sejak tahun 2016. Meredita meraih gelar sarjana komputer dari Universitas Gadjah Mada dan gelar master dari University of Nottingham. Di program studi Ilmu Komputer Meredita mengajar Pengantar Teknologi Informasi dan Algoritma, Dasar Pemrograman dan Rekayasa Perangkat Lunak.

Prakata

Bidang pemrosesan bahasa alami telah mengalami banyak perubahan selama beberapa dekade terakhir. Awalnya, pemrosesan bahasa alami menggunakan pendekatan *rule-based* dimana seseorang dapat menentukan suatu aturan jika menemukan kata ”bagus” maka kalimat dianggap memiliki sentimen yang positif. Kemudian beralih menggunakan pendekatan probabilistik yang memiliki performa lebih baik. Meskipun memiliki performa lebih baik, pendekatan ini membutuhkan banyak pekerjaan manual untuk memisahkan bagian-bagian mana dari proses yang bergantung pada *machine learning* dan *deep learning*. Baru-baru ini, dengan semakin mumpuninya infrastruktur komputer, kita dapat melatih sistem *end-to-end* yang tidak mungkin dilakukan beberapa tahun lalu. Kini kita dapat menangkap pola yang lebih kompleks dan dapat menggunakan model-model ini dalam berbagai aplikasi seperti *sentiment analysis*, *question-answering*, dan *chatbots*.

Buku ini terbagi menjadi empat bagian. Pada bagian pertama akan dibahas pendekatan sederhana menggunakan regresi logistik dan naïve bayes. Selanjutnya pada bagian kedua akan dibahas pendekatan probabilistik. Bagian ketiga membahas model sekuensial dan terakhir model attention yang merupakan model terkini di bidang pemrosesan bahasa alami.

Banyak dari aplikasi-aplikasi yang ada dalam bidang pemrosesan bahasa alami menggunakan model *attention*. Beberapa tahun lalu, membutuhkan waktu mingguan bahkan bulanan untuk melatih model untuk berbagai aplikasi dalam pemrosesan bahasa alami. Namun, dengan *attention*, model pemrosesan bahasa alami dapat dilatih hanya dalam beberapa jam. Materi dalam buku ini mengajak pembaca untuk membangun berbagai aplikasi pemrosesan bahasa alami dari dasar termasuk teknologi yang sama dengan yang digunakan dalam banyak sistem komersial besar.

Daftar Isi

Pemrosesan Bahasa Alami Menggunakan *Vector Space*

BAB 1

Analisis Sentimen Menggunakan Regresi Logistik

Tujuan:

- Memahami cara merepresentasikan teks menjadi angka.
- Memahami cara mengekstraksi fitur dari teks.
- Memahami cara mengklasifikasikan data teks menggunakan klasifier sederhana.

Pada bagian pertama ini kita akan membahas tentang klasifikasi dan *vector space* dan bagaimana mengaplikasikannya untuk menyelesaikan permasalahan seperti analisis sentimen dan penerjemah. Misalnya, kita mendapat 10.000 ulasan terhadap suatu produk dimana setiap potongan teks dituliskan oleh pelanggan. Kita akan mempelajari bagaimana membedakan mana ulasan yang positif dan negatif terhadap produk tersebut.

Sebelum dapat membedakan ulasan positif dan negatif, kita akan terlebih dahulu membahas bagaimana merepresentasikan teks sebagai vektor. Selanjutnya kita akan membangun klasifier untuk membedakan apakah teks merupakan ulasan yang memiliki sentimen positif atau justru negatif. Pada bab ini, klasifier yang akan digunakan adalah Regresi Logistik.

Meskipun sederhana, regresi logistik merupakan tools penting yang digunakan dalam berbagai aplikasi di bidang pemrosesan bahasa alami. Algoritma regresi logistik bermanfaat karena kemudahannya untuk dilatih dan kemampuannya memberikan hasil yang baik sebagai *baseline*. Pada bab ini kita akan memproses data, kemudian melatih model regresi logistik, dan menguji akurasi dari model yang dilatih.

LINGO

Baseline dalam konteks *machine learning* merujuk pada metode atau model sederhana yang digunakan sebagai titik awal perbandingan. Baseline merupakan pendekatan dasar untuk memecahkan suatu masalah, seringkali menggunakan teknik yang sederhana atau tanpa kompleksitas yang tinggi.
Regresi logistik dapat digunakan sebagai patokan untuk perbandingan dengan model atau algoritma yang lebih canggih. Jika performa model yang lebih canggih tidak berhasil melebihi baseline regresi logistik, hal tersebut menunjukkan bahwa kompleksitas tambahan mungkin tidak memberikan manfaat, atau ada permasalahan terkait data.

Kita akan mempelajari langkah-langkah untuk mengimplementasikan algoritma *supervised* khususnya regresi logistik. Dalam *machine* learning, jika kita menggunakan pendekatan *supervised* kita membutuhkan input berupa fitur X dan sekumpulan label Y. Untuk mendapatkan hasil prediksi yang seakurat mungkin, kita bertujuan sebisa mungkin untuk meminimalkan *error rate* atau *cost*. Untuk mencapai tujuan tersebut, kita akan menjalankan fungsi prediksi yang menggunakan data parameter

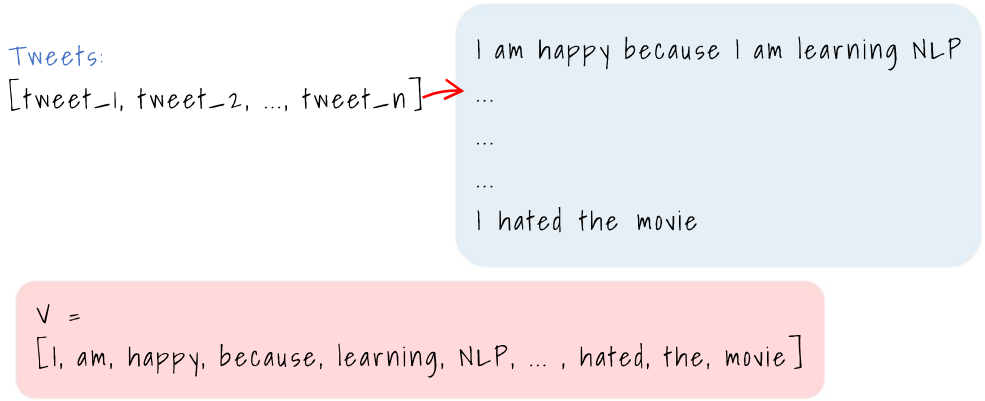
untuk memetakan fitur ke label output ($\hat{y}$). Pemetaan terbaik dari fitur ke label dicapai ketika perbedaan antara nilai yang diharapkan (y) dan nilai yang diprediksi ($\hat{y}$) diminimalkan. Hal ini dilakukan oleh *cost function* dengan membandingkan seberapa dekat output $\hat{y}$ yang kita prediksi dengan dengan label y. Setelah menghitung *cost function,* kita dapat memperbarui nilai parameter sebelumnya dan mengulangi seluruh proses ini hingga akhirnya mendapatkan nilai *cost* yang minimal.

Kita akan menggunakan contoh kasus untuk memahami cara kerja *supervised machine learning*. Misalkan kita memiliki contoh cuitan berupa kalimat "I am happy because I am learning NLP" dan *task* yang ingin dilakukan adalah mengklasifikasikan apakah cuitan tersebut memiliki sentimen positif atau negatif.
Untuk dapat melakukan klasifikasi sentimen, hal pertama yang harus kita lakukan adalah melatih sekumpulan data cuitan dimana cuitan dengan sentimen positif kita beri label 1 sebaliknya cuitan dengan sentimen negatif kita beri label 0. Untuk *task* ini kita akan menggunakan klasifier regresi logistik untuk membedakan kedua kelas ini (positif dan negatif).

Untuk membangun model menggunakan regresi logistik, pertama kita akan memproses cuitan pada dataset latih dan mengekstrak fitur-fitur yang bermanfaat. Kemudian kita akan melatih model regresi logistik untuk meminimalkan *cost* sehingga akhirnya kita akan dapat melakukan prediksi. Pada sub bab berikutnya kita akan membahas tahapan ini satu per satu.

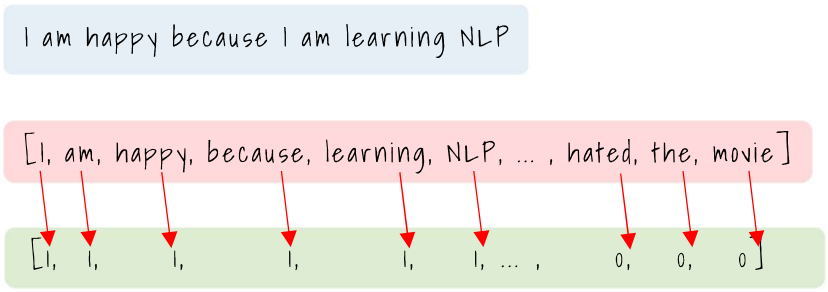
1.1 Kosakata dan Ekstraksi Fitur

Sekarang kita akan mempelajari bagaimana cara merepresentasikan sebuah kata sebagai vektor. Untuk melakukan hal tersebut, pertama kita perlu membuat kosakata (*vocabulary*). Setelah memiliki kosakata kita dapat melakukan pengkodean berbagai teks atau berbagai cuitan sebagai larik berisi angka (*array of numbers*).



Gambar 1.1 Ilustrasi Kosakata dari Kumpulan Cuitan

Misalkan kita memiliki sekumpulan cuitan seperti ditunjukkan pada Gambar 1.1, maka kita akan memiliki kosakata V yang berisi daftar kata-kata unik dari kumpulan cuitan tersebut. Untuk mendaftarkan daftar kata-kata unik, kita harus menelusuri satu per satu kata dalam kumpulan cuitan dan menyimpan setiap kata baru yang ditemukan. Pada contoh tersebut, kita akan memiliki kata I, am, happy, because, dan seterusnya. Perlu diperhatikan, walaupun kata I dan am muncul lebih dari satu kali pada cuitan, kedua kata ini hanya satu dalam kosakata.



Gambar 1.2 Ilustrasi Ekstraksi Fitur dalam Bentuk *Sparse Representation*

Selanjutnya, kita akan menggunakan kumpulan cuitan yang sama dan mengekstrak fitur menggunakan kosakata yang sudah kita miliki. Untuk melakukan hal tersebut, kita harus memeriksa jika seluruh kata dari kosakata muncul dalam cuitan. Jika kata tersebut muncul dalam cuitan, contohnya kata "I", kita akan memberikan nilai 1 untuk fitur tersebut. Sebaliknya jika kata tidak muncul maka kita akan memberikan nilai 0 untuk fitur tersebut. Pada contoh ini, dapat dilihat pada Gambar 1.2 representasi dari cuitan yang kita miliki akan memiliki enam nilai satu dan banyak nilai nol. Nilai nol yang banyak ini berkorespondensi dengan setiap kata unik dalam kosakata yang tidak muncul dalam cuitan. Bentuk representasi dengan sejumlah kecil nilai bukan nol (*non-zero values)* ini disebut sebagai *sparse representation*.

Sparse representation akan memiliki fitur sebanyak ukuran kosakata yang kita miliki. Semakin besar kosakata yang kita miliki semakin banyak juga fitur yang akan bernilai nol pada setiap cuitan. Dengan menggunakan *sparse representation,* model regresi logistik harus mempelajari n+1 parameter, dimana n adalah ukuran dari kosakata. Bisa dibayangkan, jika ukuran kosakata sangat besar, hal ini akan menimbulkan masalah. Akan membutuhkan waktu yang lama untuk melatih model dan melakukan prediksi.

Sparse representation seperti yang ditunjukkan pada Gambar 1.12 disebut juga dengan one-hot encoding. Selain one-hot encoding masih ada berbagai reprentasi sparse lainnya. Dalam buku ini yang dimaksud dengan sparse representation adalah representasi menggunakan one-hot encoding.

untuk merepresentasikan sebuah kata dalam kalimat, yang pertama kali dilakuan adalah membuat kosakata (vocabulary). Ada juga yang menyebutnya kamus (dictionary). Kosakata atau kamus adalah daftar kata-kata yang akan digunakan dalam representasi. Misalkan kita memiliki sebuah kamus yang berisi kata seperti dibawah ini:

abah	1
acar	2
...	...
dan	578
...	...
harry	4075
....	....
potter	6830
...	...
sihir	8970
...	...
zebra	10 000

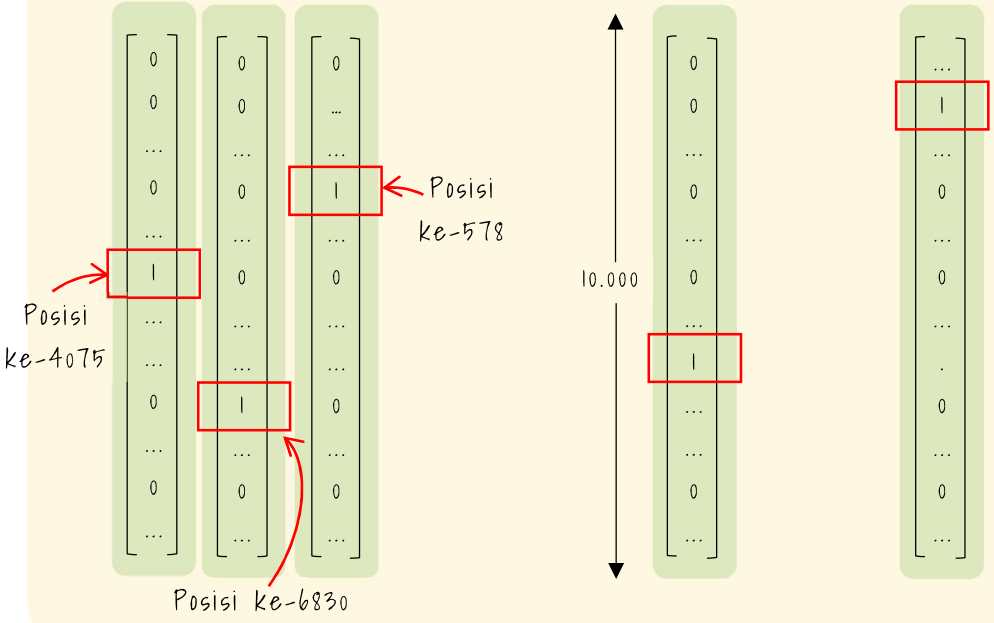
Abah adalah kata pertama, acar kata kedua, dan kata ke-578, harry kata ke-4075 hingga zebra adalah kata ke-10.000 dalam kamus

ukuran kamus adalah 10.000

Pada contoh ini kita menggunakan ukuran kamus yang relatif kecil. ukuran kamus yang biasa digunakan pada aplikasi komersil bisa mencapai jutaan kata. Umumnya ukuran yang digunakan antara 30.000 hingga 50.000

Setelah memiliki kamus, selanjutnya kita akan menggunakan representasi one-hot untuk merepresentasikan setiap kata. Misalnya, vektor yang merepresentasikan kata Harry akan berupa sebuah vektor dengan semua nilai adalah 0 kecuali pada posisi 4075 yang akan bernilai 1. Kemudian vektor yang merepresentasikan kata Potter akan berupa sebuah vektor dengan semua nilai adalah 0 kecuali pada posisi 6830 yang akan bernilai 1. Semua kata akan berupa vektor 10.000 dimensi jika kamus berisi 10.000 kata. Maka dalam representasi ini, untuk setiap kata dalam kalimat akan berupa one-hot vector. Disebut one-hot karena hanya ada tepat satu nilai 1 dan sisanya adalah 0 untuk setiap kata dalam kalimat. Pada contoh diatas, kita akan memiliki sembilan one-hot vector yang merepresentasikan sembilan kata dalam kalimat.

X : Harry Potter dan Hermione Granger menemukan mantra sihir baru.



Jika kita memiliki daftar cuitan seperti dibawah ini, berap parameter yang harus dipelajari model regresi logistik dengan iput yang direpresentasi menggunakan sparse representation?:

Hint: Ingat untuk menambahkan 1, karena adanya bias

["I am happy because I am learning NLP", "I hated that movie", I love working at Google"]

a. 18 b. 14 c. 42 d. 4

Jawaban: b

Cara lain untuk mengekstrasi fitur adalah dengan menggunakan frekuensi positif dan frekuensi negatif. Sekarang kita akan mempelajari cara melakukan perhitungannya. Misalkan kita diberikan sebuah kata, kita ingin menelusuri berapa kali kata tersebut muncul sebagai kelas positif. Jika ada kata lain, kita ingin menelusuri berapa kali kata tersebut muncul sebagai kelas negatif. Menggunakan kedua perhitungan tersebut, kita kemudian dapat mengekstrak fitur dan menggunakan fitur-fitur tersebut sebagai input untuk klasifier.