SKRIPSI

SISTEM PENILAIAN KEPUASAN PELANGGAN TERHADAP TRANSAKSI PEMBELIAN PRODUK E-COMMERCE MENGGUNAKAN ANALISIS SENTIMEN

EUGENIUS WAHYUDIARTO

D121171526

DEPARTEMEN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS HASANUDDIN
MAKASSAR
2022

SKRIPSI

SISTEM PENILAIAN KEPUASAN PELANGGAN TERHADAP TRANSAKSI PEMBELIAN PRODUK E-COMMERCE MENGGUNAKAN ANALISIS SENTIMEN

TUGAS AKHIR

Sebagai salah satu syarat Mencapai Gelar Sarjana Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin Makassar

Disusun dan diajukan oleh:

EUGENIUS WAHYUDIARTO

D121171526

kepada

DEPARTEMEN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS HASANUDDIN
MAKASSAR
2022

LEMBAR PENGESAHAN SKRIPSI

SISTEM PENILAIAN KEPUASAN PELANGGAN TERHADAP TRANSAKSI PEMBELIAN PRODUK E-COMMERCE MENGGUNAKAN ANALISIS SENTIMEN

Disusun dan diajukan oleh

EUGENIUS WAHYUDIARTO

D121171526

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Teknik Informatika Fakultas Teknik Universitas Hasanuddin pada tanggal 20 April 2022 dan dinyatakan telah memenuhi syarat kelulusan.

Menyetujui,

Pembimbing Utama,	Pembimbing Pendamping,
Dr. Amil Ahmad Ilham, ST.,MIT.	Anugrayani Bustamin, S.T., M.T.
Nip. 197310101998021001	Nip. 199012012018074001

Ketua Program Studi,

Dr. Amil Ahmad Ilham, S.T., M.IT.

Nip. 19731010 199802 1 002

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama: Eugenius Wahyudiarto

NIM: D121171526

Departemen: Teknik Informatika

Jenjang: S1

Menyatakan dengan ini bahwa karya tulisan saya yang berjudul

SISTEM PENILAIAN KEPUASAN PELANGGAN
TERHADAP TRANSAKSI PEMBELIAN
PRODUK E-COMMERCE
MENGGUNAKAN ANALISIS SENTIMEN

Adalah karya tulisan saya sendiri dan bukan merupakan pengambil alihan tulisan orang lain bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri.

Apabila di kemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Makassar, 23 Mei 2022

Yang Menyatakan

(Eugenius Wahyudiarto)

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena berkat rahmat dan karunia-Nya sehingga skripsi yang berjudul **"Sistem Penilaian Kepuasan Pelanggan Terhadap Transaksi Pembelian Produk E-commerce Menggunakan Analisis Sentimen"** ini dapat terselesaikan dengan baik sebagai salah satu syarat dalam menyelesaikan jenjang Strata-1 pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Penulis menyadari bahwa banyak kendala yang dihadapi dalam menyelesaikan skripsi ini. Berkat dukungan, bimbingan, serta motivasi yang diberikan, penyusunan skripsi ini dapat terselesaikan dengan baik. Ucapan terima kasih serta penghargaan yang setinggi-tingginya penulis sampaikan kepada:

1. Tuhan Yang Maha Esa yang melalui berkat dan rahmat-Nya sehingga penulis dapat menyelesaikan tugas akhir ini.
2. Kedua orang tua penulis, Bapak Yulius Tandiayu, dan Ibu Anastasia Sunarseh yang selalu memberikan dukungan, doa, semangat dan kasih sayang serta selalu sabar dalam mendidik penulis.
3. Harrison Kinsley & Daniel Kukieła yang telah membuat materi *Neural Networks from Scratch in Python* sehingga membantu penulis untuk lebih memahami secara mendalam secara teori dan praktek mengenai *Neural Network*. Khususnya, Harrison Kinsley atau Sentdex di Youtube

yang juga membimbing penulis untuk menulis program menggunakan Python.

4. Younes Bensouda Mourri sebagai AI instruktur di Stanford University dan Łukasz Kaiser sebagai Staff Research Scientist, Google Brain & Chargé de Recherche, CNRS, yang telah membimbing penulis dalam memahami NLP melalui materinya di Natural Language Processing Specialization.
5. Bapak Dr. Amil Ahmad Ilham, S.T., M.IT., selaku pembimbing I dan ketua Prodi Teknik Informatika Fakultas Teknik Universitas Hasanuddin, serta Ibu Anugrahyani Bustamin, S.T., M.T., selaku pembimbing II yang selalu menyediakan waktu, tenaga, pikiran dan memberikan bimbingan dalam penyusunan skripsi ini.
6. Bapak Ir. Christiforus Yohannes, M.T., selaku penguji I, serta Bapak Dr. Muhammad Niswar, S.T., M.T., selaku penguji II yang memberikan saran dan pelajaran yang sangat berharga dalam hidup penulis.
7. Fitriani Nasir, S.T., yang memberikan arahan mendetail dan praktis bagi penulis dalam menyelesaikan tugas akhir ini.
8. Ayu Lestari, S.Si., yang turut membantu penulis mengoreksi skripsi ini.
9. Teman-teman Prodi Teknik Informatika angkatan 2017, 2021 atas dukungan dan semangat yang telah diberikan.
10. Segenap Dosen dan Staff Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah banyak membantu penulis selama masa perkuliahan.

11. Serta berbagai pihak atas segala dukungan dan bantuannya yang tidak dapat penulis tuliskan satu persatu.

Penulis berharap semoga Tuhan Yang Maha Esa membalas segala kebaikan dari semua pihak yang telah membantu penulis dalam penyusunan skripsi ini dan semoga skripsi ini dapat memberikan hal yang bermanfaat serta menambah wawasan ilmu untuk pembaca dan juga bagi penulis sendiri.

Akhir kata, penulis mengucapkan selamat membaca dan mendalami skripsi ini. Penulis akan membawa pembaca dalam perjalanan untuk mencapai tujuan skripsi ini. Selanjutnya untuk menghubungi penulis, pembaca dapat mengirimkan pesan melalui eugeniusw18@gmail.com. Berbahagialah orang yang haus akan pengetahuan, karena dia akan tahu.

Makassar, Februari 2022

Penulis,

(Eugenius Wahyudiarto)

ABSTRAK

Saat ini semakin banyak produk bermunculan. Berbagai layanan yang menawarkan produk yang mirip dan sama membuat pembeli sulit memutuskan untuk membeli sebelum melihat ulasan dari pengguna lainnya. Hal ini diperparah dengan tumbuhnya platform pemasaran (*e-commerce*) yang berbeda-beda. Pengguna pun harus menghabiskan waktu untuk memilih produk di masing-masing platform dengan banyak pertimbangan, seperti melihat *rating*, harga, dan ulasan dari pembeli lainnya. Optimalisasi pemilihan produk *e-commerce* dilakukan supaya pengguna tidak harus menghabiskan waktu yang lama dengan membaca setiap ulasan ketika ingin membeli produk. Dengan adanya sistem penilaian kepuasan pelanggan terhadap transaksi pembelian produk e-commerce menggunakan analisis sentimen diharapkan dapat memberikan penilaian menyeluruh terhadap transaksi pembelian suatu produk dari ulasan yang diberikan. Sistem ini memberikan 91% akurasi dan skor F_1 sebesar 57% dalam memprediksi sentimen menggunakan naive Bayes. Model naive Bayes kemudian digunakan untuk mendapatkan skor kepuasan pelanggan terhadap suatu produk.

Kata kunci: skor kepuasan pelanggan**,** rekomendasi pembelian produk, analisis sentimen, naive Bayes.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar belakang

Saat ini semakin banyak produk bermunculan. Bahkan di antara produk tersebut terdapat banyak kemiripan dan kesamaan di dalamnya sehingga pembeli sulit memutuskan di mana pembelian produk yang paling cocok untuk mereka. Hal ini diperparah dengan tumbuhnya platform pemasaran (*e-commerce*) yang berbeda-beda. Pengguna pun harus menghabiskan waktu untuk memilih produk di masing-masing platform dengan banyak pertimbangan, seperti melihat *rating*, harga, dan ulasan dari pembeli lainnya.

Ini adalah masalah rekomendasi. Terdapat dua kategori teknik rekomendasi, *content based filtering* (CBF) dan *collaborative filtering* (CF) (Shahbazi & Byun, 2020). Ada juga yang menggabungkan keduanya sehingga menjadi *hybrid*. CBF akan merekomendasikan produk berdasarkan kemiripannya dengan yang telah dibeli sebelumnya dan CF akan merekomendasikan produk berdasarkan yang dibeli entitas lain. Ada beberapa solusi yang dapat dilakukan untuk mengatasi masalah di atas. Pertama, fitur pencarian produk yang lebih baik. Fitur pencarian ini akan masuk dalam model klasifikasi tag dan melakukan *query* ke database. Kedua, merekomendasikan produk yang paling baik. Untuk mendapatkan produk yang paling baik perlu dilakukan perbandingan antara produk dan juga *review* dari setiap produk. *Sentiment analysis* (SA) akan digunakan

untuk mengetahui sikap pengguna terhadap produk yang ditawarkan (Khanvilkar & Deepali Vora, 2018). Terdapat beberapa algoritma yang dapat digunakan dalam SA seperti XGBoost, Random Forest, SVM, KNearest Logistic Regression (Shahbazi & Byun, 2020), dan naive Bayes (Parveen & Pandey, 2016). Bahkan ada yang menggunakan *lexicon-based keyword* yang digabungkan dengan algoritma tersebut (Khanvilkar & Deepali Vora, 2018). Tantangannya adalah *preprocessing* datanya seperti, kesalahan ketik dari pengguna, singkatan Bahasa Indonesia dan *emoticon*. Karena *emoticon* memainkan peranan penting dalam SA (Parveen & Pandey, 2016).

Sistem yang akan dibuat seperti yang dilakukan oleh Kim (Kim dkk., 2019). Perbedaannya adalah sistem yang dibuat oleh Kim hanya menggunakan *keyword* kata positif dan negatif untuk menentukan sentimennya. Sistem yang dibuat oleh Hanni dapat membandingkan spesifikasi produk (Hanni dkk., 2016). Sayangnya tidak semua produk memiliki spesifikasi. Alternatifnya adalah membuat skor penilaian kepuasan pelanggan, sebagai rekomendasi, berdasarkan hasil SA sehingga mendapatkan ringkasan skor mengenai apa yang dirasakan pelanggan ketika melakukan transaksi pembelian produk terkait.

1.2 Rumusan masalah

Adapun rumusan masalah dalam penelitian ini yaitu, bagaimana mengembangkan model analisis sentimen dalam penilaian kepuasan

pelanggan terhadap transaksi produk e-commerce dengan menggunakan algoritma naive Bayes?

1.3 Tujuan penelitian

Tujuan dari penelitian ini adalah untuk mengetahui bagaimana mengembangkan model analisis sentimen menggunakan algoritma naive Bayes dalam penilaian kepuasan pelanggan terhadap transaksi pembelian produk e-commerce.

1.4 Batasan penelitian

1. Data ulasan produk diambil dari Tokopedia sebanyak minimal 3 kategori dan jumlah data minimal 50000 per kategori.
2. Model analisis sentimen menggunakan naive Bayes yang dibuat dengan pemrograman Python versi 3.7.
3. Sistem penilaian ini berdasarkan ulasan dari pelanggan.

1.5 Manfaat penelitian

1. Penelitian dapat memberikan sumbangan pemikiran mengenai penilaian kepuasan pelanggan terhadap transaksi pembelian produk e-commerce.
2. Penelitian dapat mempermudah masyarakat dalam pembelian produk tanpa harus menghabiskan waktu untuk membaca ulasan setiap produk.
3. Penelitian dapat merepresentasikan skor kepuasan pelanggan dengan lebih akurat dibandingkan dengan menggunakan bintang dari pelanggan yang kadangkala tanpa komentar apa-apa.

4. Penelitian ini dapat diterapkan dalam kasus lain yang sejenis, misalnya penilaian video berdasarkan komentar atau penilaian aplikasi berdasarkan ulasan.

1.6 Sistematika penulisan

Sebagai gambaran singkat dari isi tulisan ini, penulis memberikan uraian singkat pada masing-masing tahapan penelitian.

BAB I PENDAHULUAN

Pada tahap ini dijabarkan latar belakang, rumusan masalah, tujuan, batasan, manfaat, dan sistematika penulisan dari penelitian ini.

BAB II TINJAUAN PUSTAKA

Pada tahap ini dituliskan referensi-referensi istilah dan metode yang terkait dengan sistem penilaian kepuasan pelanggan terhadap transaksi produk e-commerce menggunakan analisis sentimen.

BAB III METODOLOGI PENELITIAN

Pada tahap ini dijelaskan secara detail penelitian yang dilakukan. Beberapa poin pembahasannya yaitu, tempat dan waktu, alat dan bahan, metode penelitian, rancangan sistem, serta analisis sistem.

BAB IV HASIL DAN PEMBAHASAN

Pada tahap ini dijelaskan tentang hasil dan pembahasan dari penelitian yang dilakukan.

BAB V PENUTUP

Pada tahap ini diberikan kesimpulan yang didapatkan dari penelitian yang telah dilakukan. Terdapat juga saran-saran untuk pengembangan selanjutnya.

BAB II
TINJAUAN PUSTAKA

2.1 Analisis sentimen

Sentimen adalah pandangan atau perasaan terhadap suatu peristiwa atau sesuatu. Analisis adalah pemeriksaan secara teliti terhadap suatu hal. Analisis sentimen atau s*entiment analysis* bertujuan untuk secara otomatis mengungkap sikap mendasar yang kita pegang terhadap suatu entitas. Analisis sentimen memiliki banyak aplikasi salah satunya adalah mengukur tingkat kepuasan pelanggan terhadap suatu merek terhadap merek lainnya. Saat ini analisis sentimen yang secara luas digunakan berbasis teks (Soleymani dkk., 2017, hlm. 2).

2.2 Web scraper

Web scraper adalah suatu perangkat lunak yang menyimulasikan kunjungan manusia pada web untuk mengumpulkan data informasi terperinci. Keuntungan menggunakan *scraper* yakni dapat diprogram dan diotomatisasikan sehingga pengumpulan informasi menjadi lebih cepat dan informasi dapat disimpan pada format yang terstruktur (Diouf dkk., 2019, hlm. 6040).

2.3 *Numeric character references*

Numeric character references menentukan posisi kode karakter dalam kumpulan karakter dokumen. Referensi karakter numerik dapat mengambil dua bentuk:

1. Sintaks "**&#D;**". D adalah angka desimal, mengacu pada angka karakter desimal ISO 10646.
2. Sintaks "**&#xH;**" atau "**&#XH;**". H adalah angka heksadesimal, mengacu pada nomor karakter heksadesimal ISO 10646. Angka heksadesimal dalam referensi karakter numerik tidak peka huruf besar-kecil.

Berikut ini beberapa contoh representasi *numeric character references*:

1. **&#29;** (dalam desimal) merepresentasikan karakter apostrof.
2. **"**, **"**, **",** atau **"** (dalam heksadesimal) merepresentasikan karakter apostrof (World Wide Web Consortium / W3C, 2018, Bab 5).

2.4 Google Cloud Natural Language API

Google Cloud Natural Language (GCloud NL) API adalah antarmuka pemrograman aplikasi dari Google yang dikembangkan dengan menggunakan model yang telah dilatih sebelumnya. Model ini memberdayakan pengembang untuk dengan mudah menerapkan Natural Language Understanding (NLU) ke aplikasi mereka dengan fitur-fitur termasuk analisis sentimen, analisis entitas, analisis sentimen entitas, klasifikasi konten, dan analisis sintaksis (Google LLC, 2021a).

Berikut ini contoh fungsi untuk menganalisa sentimen suatu teks menggunakan Google Cloud dalam bahasa pemrograman Python dari https://cloud.google.com/natural-language/docs/analyzing-sentiment.

```
from google.cloud import language_v1

def sample_analyze_sentiment(text_content):
    """
    Analyzing Sentiment in a String

    Args:
      text content The text content to analyze
    """

    client = language_v1.LanguageServiceClient()

    # text_content = 'I am so happy and joyful.'

    # Available types: PLAIN_TEXT, HTML
    type_ = language_v1.Document.Type.PLAIN_TEXT

    # Optional. If not specified, the language is automatically detected.
    # For list of supported languages:
    # https://cloud.google.com/natural-language/docs/languages
    language = "en"
    document = {"content": text_content, "type_": type_, "language": language}

    # Available values: NONE, UTF8, UTF16, UTF32
    encoding_type = language_v1.EncodingType.UTF8

    response = client.analyze_sentiment(request = {'document': document,
            'encoding_type': encoding_type})
    # Get overall sentiment of the input document
    print(u"Document sentiment score: {}".format(response.document_sentiment.score))
    print(
        u"Document sentiment magnitude: {}".format(
            response.document_sentiment.magnitude
        )
    )
    # Get sentiment for all sentences in the document
    for sentence in response.sentences:
        print(u"Sentence text: {}".format(sentence.text.content))
        print(u"Sentence sentiment score: {}".format(sentence.sentiment.score))
        print(u"Sentence sentiment magnitude: {}".format(sentence.sentiment.magnitude))

    # Get the language of the text, which will be the same as
    # the language specified in the request or, if not specified,
    # the automatically-detected language.
    print(u"Language of the text: {}".format(response.language))
```

Sentimen dari suatu teks dapat dinterpretasikan dengan melihat nilai *score* dan *magnitude*-nya. Google Cloud memberikan contoh pada **Gambar 2.4.1** untuk untuk menginterpretasikannya.

Sentiment	Sample Values
Clearly Positive*	`"score"`: 0.8, `"magnitude"`: 3.0
Clearly Negative*	`"score"`: -0.6, `"magnitude"`: 4.0
Neutral	`"score"`: 0.1, `"magnitude"`: 0.0
Mixed	`"score"`: 0.0, `"magnitude"`: 4.0

Gambar 2.4.1 Contoh interpretasi sentimen
Sumber: (Google LLC, 2021b, bag. Natural Language API Basics)

Nilai *score* adalah skor sentimen mulai dari -1.0 (negatif) sampai 1.0 (positif). Nilai *magnitude* menandakan kekuatan emosional (baik itu positif maupun negatif) dalam teks mulai dari 0.0 sampai +inf. Seandainya nilai *score* adalah 0 bukan berarti itu adalah netral. Bisa saja tingkat emosional negatif dan positif adalah sama sehingga harus dilakukan pengecekan nilai *magnitude*. Penentuan c*learly positive* dan *negative* bervariasi tergantung kasus penggunaan. Misalnya saja dapat didefinisikan *threshold* skor di atas 0.25 sebagai *clearly positive*. Namun setelah dilihat hasilnya skor 0.15 masih positif maka *threshold* dapat diubah (Google LLC, 2021b).

2.5 Algoritma Naive Bayes

Algoritma naive Bayes adalah algoritma probabilitas berdasarkan Teorema Bayes untuk menyelesaikan masalah klasifikasi (Kathuria, 2020; Zhang, 2019). Teorema Bayes adalah cara untuk mencari suatu probabilitas ketika kita mengetahui probabilitas lainnya. Persamaannya adalah pada persamaan **(1)** sebagai berikut (Pierce, 2020; Zhang, 2019).

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \quad \textbf{(1)}$$

P(Y|X) / *posterior* : seberapa sering Y terjadi jika X terjadi;

P(X|Y) / *likelihood* : seberapa sering X terjadi jika Y terjadi;

P(Y) / *prior* : seberapa besar kemungkinan Y terjadi;

P(X) / *evidence* : seberapa besar kemungkinan X terjadi;

Dalam skenario nyata, X atau fitur tidak hanya satu dan selalu ada ketergantungan di antaranya. Misalkan terdapat 5 fitur sehingga secara naif persamaan **(1)** dimodifikasi menjadi persamaan naive Bayes di persamaan **(2)** berikut yang mengasumsikan semua fitur independen.

$$P(y|x_1, x_2, x_3, x_4, x_5) = \frac{P(x_1|y)P(x_2|y)P(x_3|y)P(x_4|y)x_5|y) * P(y)}{P(x_1)P(x_2)P(x_3)P(x_4)P(x_5)} \quad \textbf{(2)}$$

Persamaan **(2)** juga dapat dituliskan sebagai persamaan **(3)** berikut.

$$P(y|x_1, x_2, x_3, x_4, x_5) = \frac{P(y)\Pi_{i=1}^{5} P(x_i|y)}{P(x_1)P(x_2)P(x_3)P(x_4)P(x_5)} \quad \textbf{(3)}$$

Perhatikan bahwa denominator pada persamaan **(3)** akan konstan. Sehingga secara proporsi persamaan **(3)** akan menjadi persamaan **(4)** berikut.

$$P(y|x_1, x_2, x_3, x_4, x_5) \propto P(y)\Pi_{i=1}^{5} P(x_i|y) \quad \textbf{(4)}$$

Ini akan memberikan kemungkinan untuk setiap kelas y. Yang lebih tinggi dianggap sebagai prediksinya. Sehingga untuk mendapatkan nilai tersebut digunakan argmax seperti pada persamaan **(5)** (Kathuria, 2020).

$$\hat{y} = \arg\max_{y \in Y} P(y)\Pi_{i=1}^{5} P(x_i|y) \quad \textbf{(5)}$$

Untuk mengaplikasikan naive Bayes pada klasifikasi teks, X di sini menjadi kata dalam dokumen. Perhitungan naive Bayes dilakukan di ruang log untuk menghindari *underflow* (kalkulasi yang terlalu kecil untuk direpresentasikan oleh CPU atau memori komputer) dan meningkatkan

kecepatan komputasi. Sehingga persamaan **(5)** dapat direpresentasikan sebagai persamaan **(6)** berikut.

$$\hat{y} = \arg\max_{y \in Y} logP(y) + \sum_{i=1}^{5} logP(x_i|y) \quad \textbf{(6)}$$

Dengan menggunakan ruang log, persamaan **(6)** akan memprediksi kelas sebagai fungsi linear terhadap fitur yang diberikan (Jurafsky & Martin, 2020).

2.6 Pelatihan Classifier Naive Bayes

Pelatihan model klasifikasi sentimen dengan menggunakan persamaan **(6)**, misalkan $y=c$ dan $x_i=w_i$, dilakukan dengan mencari nilai *P(c)* dan $P(w_i|c)$. *Prior* atau *P(c)* adalah berapa persen setiap kelas sentimen *c* dalam data latih. Untuk menentukannya digunakan persamaan **(7)** berikut.

$$\hat{P}(c) = \frac{N_c}{N_{doc}} \quad \textbf{(7)}$$

$\boldsymbol{N_c}$: banyaknya data latih dengan kelas sentimen *c;*

$\boldsymbol{N_{doc}}$: banyaknya data latih

Likelihood atau $P(w_i|c)$ diasumsikan sebagai rasio antara jumlah kemunculan suatu kata dalam *bag of words* dokumen terhadap jumlah kata dengan kelas *c*, yang dihitung dengan persamaan **(8)** berikut.

$$\hat{P}(w_i|c) = \frac{count(w_i, c)}{\sum_{w' \in V} count(w', c)} \quad \textbf{(8)}$$

V : terdiri dari semua kata dari semua kelas *c;*

count(w_i, c) : jumlah kata dengan kelas *c*, contoh: “fantastic”=1;

count(w’, c) : jumlah semua kata dalam kelas *c*, contoh: 100 kata positif dari 1000 kata (1/10);

Tetapi apabila suatu kata tidak muncul dalam data latih, misalnya kata “fantastic” maka nominator akan bernilai 0 sehingga seluruh persamaan bernilai 0.

$$\hat{P}(\text{"}fantastic\text{"}|positive) = \frac{count(\text{"}fantastic\text{"}, positive)}{\sum_{w' \in V} count(w', positive)} = 0 \quad \textbf{(9)}$$

Hal ini menjadi masalah karena naive Bayes akan mengalikan semua fitur *likelihoods* bersama sehingga satu saja 0 akan membuat nilai *likelihoods* dari fitur lain menjadi 0. Ini dapat diatasi dengan menambahkan 1 (*Laplace smoothing*), seperti pada persamaan **(10)** berikut.

$$\hat{P}(w_i|c) = \frac{count(w_i, c) + 1}{\sum_{w' \in V}(count(w', c) + 1)} = \frac{count(w_i, c) + 1}{(\sum_{w' \in V} count(w', c)) + |V|} \quad \textbf{(10)}$$

Untuk menyelesaikan masalah negasi pada sentimen, dapat ditambahkan kata “NOT_” di depan setiap kata setelah kata negasi. Penggunaan *stopwords* tidak meningkatkan performa model, sehingga algoritma naive Bayes pada **Gambar 2.6.1** berikut tidak mengikutsertakan *stopwords* (Jurafsky & Martin, 2020).

function TRAIN NAIVE BAYES(D, C) **returns** log $P(c)$ and log $P(w|c)$

for each class $c \in C$ # Calculate $P(c)$ terms
  N_{doc} = number of documents in D
  N_c = number of documents from D in class c
  $logprior[c] \leftarrow \log \frac{N_c}{N_{doc}}$
  $V \leftarrow$ vocabulary of D
  $bigdoc[c] \leftarrow$ **append**(d) **for** d $\in$ D **with** class c
  for each word w in V # Calculate $P(w|c)$ terms
    $count(w,c) \leftarrow$ # of occurrences of w in $bigdoc[c]$
    $loglikelihood[w,c] \leftarrow \log \frac{count(w,c) + 1}{\sum_{w' \text{ in } V} (count(w',c) + 1)}$
return $logprior$, $loglikelihood$, V

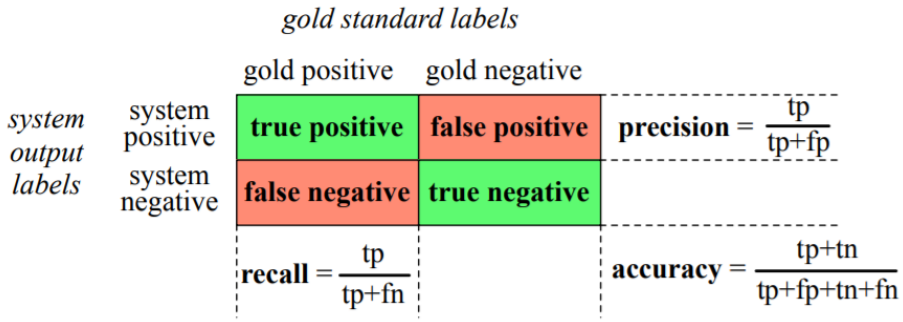
function TEST NAIVE BAYES($testdoc$, $logprior$, $loglikelihood$, C, V) **returns** best c

for each class $c \in C$
  $sum[c] \leftarrow logprior[c]$
  for each position i in $testdoc$
    $word \leftarrow testdoc[i]$
    if $word \in V$
      $sum[c] \leftarrow sum[c] + loglikelihood[word,c]$
return $\text{argmax}_c\ sum[c]$

Gambar 2.6.1 Algoritma naive Bayes untuk teks
Sumber: (Jurafsky & Martin, 2020, hlm. 60)

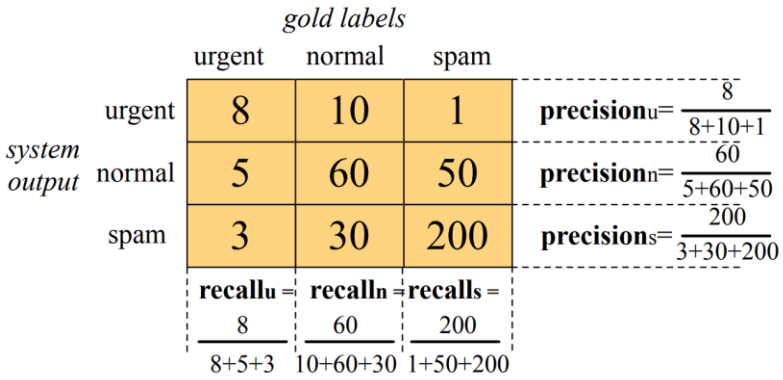
2.7 Evaluasi Model Klasifikasi

Untuk mengevaluasi model klasifikasi diperlukan pembuatan *confusion matrix* dari model tersebut. *Confusion matrix* berfungsi untuk memvisualisasikan bagaimana performa sistem terhadap label yang diberikan (*gold standard label*). Dengan *confusion matrix* dapat dihitung nilai *precision*, *recall*, *accuracy* dan pengukuran *F*. Pada **Gambar 2.7.1** berikut format *confusion matrix* pada klasifikasi biner (Jurafsky & Martin, 2020).



Gambar 2.7.1 Confusion matrix pada klasifikasi biner.
Sumber: (Jurafsky & Martin, 2020, hlm. 65)

Recall adalah rasio antara semua kelas yang diklasifikasi benar pada kelas positif terhadap total anggota aktual dari kelas positif. Dengan kata lain ini memberi tahu tentang berapa banyak dari jumlah total contoh positif yang diklasifikasikan benar. *Precision* adalah rasio antara semua anggota yang diklasifikasikan benar dalam kelas positif terhadap jumlah total anggota yang diklasifikasikan dalam kelas positif dan benar (Villalobos, 2020). Bagaimana bila ingin mengevaluasi performa model multi kelas? Untuk model multi kelas dapat menggunakan *macroaveraging*, yaitu precision dan recall dihitung seperti pada **Gambar 2.7.2** dan hasil akhirnya dirata-ratakan.



Gambar 2.7.2 Confusion matrix pada klasifikasi tiga kelas.
Sumber: (Jurafsky & Martin, 2020, hlm. 67)

Nilai *recall* dan *precision* digunakan apabila model menggunakan data yang tidak seimbang sehingga nilai *accuracy* menjadi tidak berguna. Terdapat juga persamaan *F* apabila ingin menggabungkan nilai *recall* dan *precision* dengan menggunakan parameter β sebagai nilai pementingnya pada persamaan **(11)**. Apabila β > 1 maka lebih mementingkan *recall* dan bila β < 1 maka lebih mementingkan *precision*.

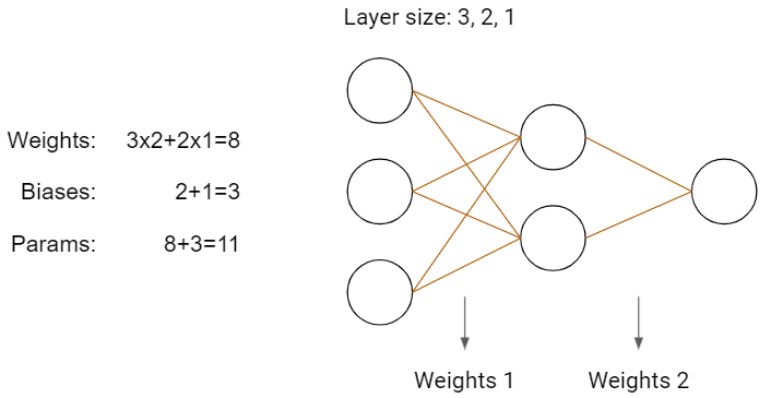
$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R} \tag{11}$$

Apabila nilai β = 1 maka nilai *recall* dan *precision* dianggap seimbang dan disebut sebagai persamaan F_1 yang dirumuskan dalam persamaan **(12)** berikut (Jurafsky & Martin, 2020).

$$F_1 = \frac{2PR}{P + R} \tag{12}$$

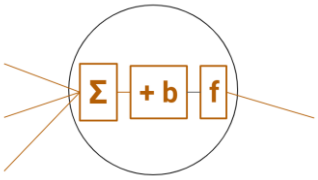
2.8 Neural Network

Artificial neural network (ANN) (yang sekarang orang-orang menyebutnya *neural network* tanpa kata "*artificial*") terinspirasi dari otak biologis, diterjemahkan ke komputer. ANN terdiri dari syaraf-syaraf (*neurons*) yang saling terhubung satu sama lain. Hal inilah yang membuat ANN seringkali mengalahkan performa metode *machine learning* yang lain. Struktur *neural network* dapat dilihat pada **Gambar 2.8.1** berikut.



Gambar 2.8.1 Neural network 3 layers

Setiap *neuron* pada neural network memiliki struktur seperti pada **Gambar 2.8.2** (Kinsley & Kukiela, 2020).



Gambar 2.8.2 Neuron pada neural network

Neural network, dalam *Weights* 1 (lihat: **Gambar 2.8.1**) atau layer(3) ke layer(2), dapat memproses dalam satu hitungan dengan menggunakan matriks pada *input* X (matriks) dan menghitung setiap *output* Y (matriks) sesuai dengan persamaan **(13)** (Aflak, 2021).

$$Y = XW + B \tag{13}$$

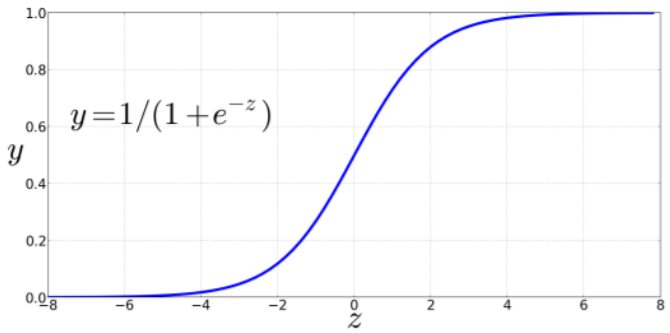
Y : Matriks prediksi;

X : Matriks input;

W : Matriks *weights*;

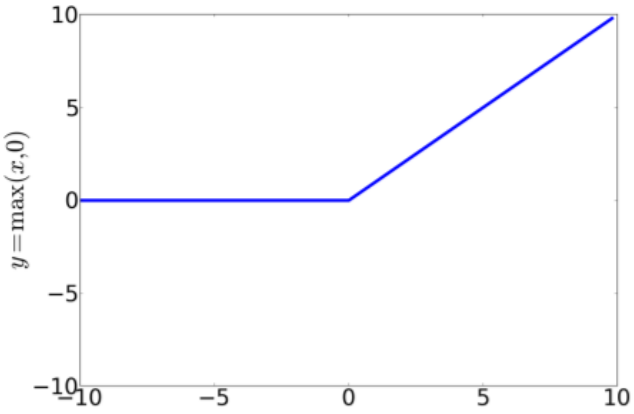
B : Matriks *bias*;

Untuk menambahkan non-linearitas pada model, perlu ditambahkan fungsi non-linear dari output beberapa *layers* (Aflak, 2021). Fungsi non-linear ini seperti sigmoid pada **Gambar 2.8.3**;



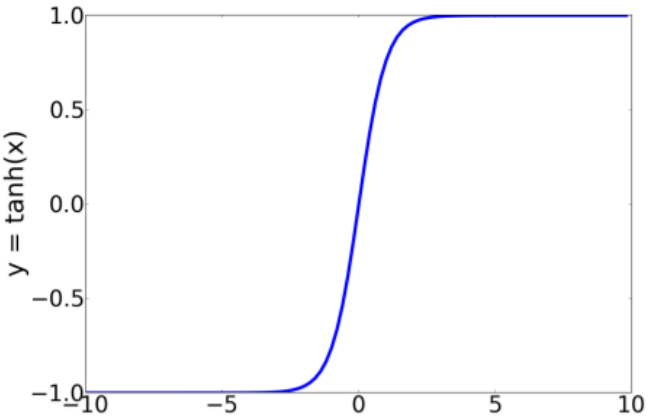
Gambar 2.8.3 Fungsi aktivasi sigmoid
Sumber: (Jurafsky & Martin, 2020, hlm. 128)

ReLu atau *rectified linear unit* pada **Gambar 2.8.4**;



Gambar 2.8.4 Fungsi aktivasi ReLu
Sumber: (Jurafsky & Martin, 2020, hlm. 130)

dan tanh pada **Gambar 2.8.5**, sebagai variasi dari sigmoid dengan jangkauan output -1 sampai 1 (Jurafsky & Martin, 2020).

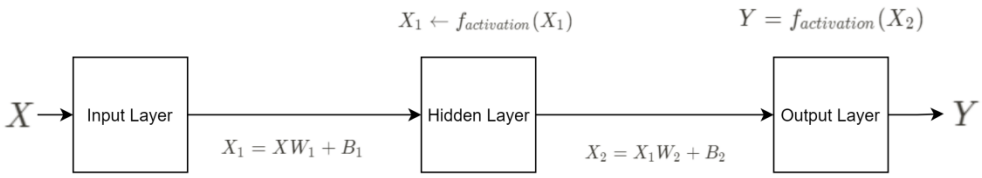


Gambar 2.8.5 Fungsi aktivasi tanh
Sumber: (Jurafsky & Martin, 2020, hlm. 130)

Fungsi aktivasi di atas dapat dituliskan secara umum sebagai persamaan **(14)** berikut (Aflak, 2021).

$$Y = f_{activation}(X) \tag{14}$$

Dengan demikian, *forward propagation* dapat dilakukan seperti pada **Gambar 2.8.6** berikut (Aflak, 2021).



Gambar 2.8.6 Proses *forward propagation*

Agar model dapat meminimalkan error antara nilai prediksi dan nilai aktual perlu dilakukan perubahan *weights* dan *biases*. Untuk dapat menentukan *weights* dan *biases* yang optimal digunakan algoritma *gradient descent* (Jurafsky & Martin, 2020). Perubahan nilai *weights* dan *biases* dari *neural network* menggunakan persamaan *gradient descent* **(15)**, **(16)**, **(17)**, dan **(18)** berikut (Aflak, 2021).

$$W \leftarrow W - \alpha \frac{\partial E}{\partial W} \tag{15}$$

$$\frac{\partial E}{\partial W} = X^T \frac{\partial E}{\partial Y} \tag{16}$$

$$B \leftarrow B - \alpha \frac{\partial E}{\partial B} \tag{17}$$

$$\frac{\partial E}{\partial B} = \frac{\partial E}{\partial Y} \tag{18}$$

Perhitungan nilai *error* antara nilai prediksi dan nilai aktual disebut sebagai fungsi *loss*, *cost*, atau, *error* dan ditentukan sendiri oleh pelatih model. Fungsi *loss* merepresentasikan total *error* dalam *neural network* (Aflak, 2021; Taylor, 2017). Terdapat beberapa fungsi *loss* seperti, *Mean Squared Error* (MSE), *Squared Error* (SE), *Root Mean Squared Error* (RMSE) dan *Sum of Squared Errors* (SSE) (Taylor, 2017). Untuk memahami *backpropagation* akan digunakan fungsi *loss* MSE dengan persamaan **(19)**:

$$E = \frac{1}{n}\Sigma_{i=1}^{n}(y_{target}^{(i)} - y^{(i)})^2 \quad \textbf{(19)}$$

Turunan dari persamaan **(19)** adalah persamaan **(20)**:

$$\frac{\partial E}{\partial Y} = [\frac{\partial E}{\partial y_1}...\frac{\partial E}{\partial y_n}] = \frac{2}{n}[y^{(1)} - y_{target}^{(1)}...y^{(n)} - y_{target}^{(n)}] = \frac{2}{n}(Y - Y_{target}) \quad \textbf{(20)}$$

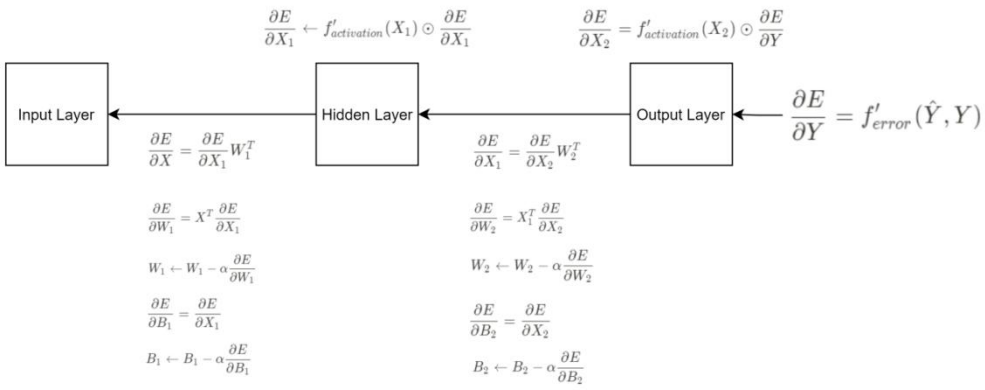
Error dari suatu *layer* dapat diteruskan ke *layer* sebelumnya dengan persamaan **(21)** berikut.

$$\frac{\partial E}{\partial X} = \frac{\partial E}{\partial Y}W^T \quad \textbf{(21)}$$

Pada fungsi aktivasi, *error* dari *layer* sebelumnya dihitung menggunakan persamaan **(22)** berikut.

$$\frac{\partial E}{\partial X} \leftarrow f'_{activation}(X) \odot \frac{\partial E}{\partial Y} \quad \textbf{(22)}$$

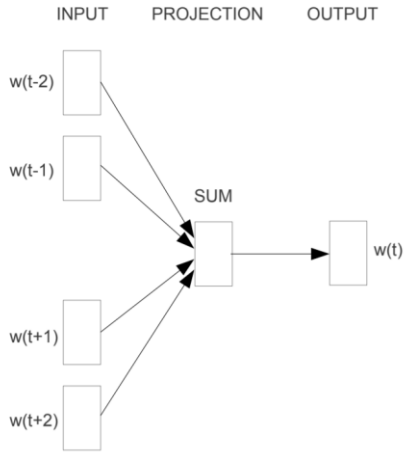
Perhatikan bahwa pada persamaan **(22)** digunakan *hadamard product*, yaitu perkalian matriks identikal secara elemen. Dengan demikian, proses *backpropagation* terjadi seperti pada **Gambar 2.8.7** (Aflak, 2021).



Gambar 2.8.7 Proses *backpropagation*

2.9 *Continuous Bag-of-Words* (CBOW)

Continuous Bag-of-Words (CBOW) adalah arsitektur yang mirip *neural network language model* (NNLM) yang mencoba memprediksi kata yang dikelilingi oleh kata-kata lain atau konteks (sebesar C dari kata tengah). CBOW merupakan salah satu arsitektur dari Word2Vec dan yang lainnya adalah skip-gram. Pada CBOW semua input direpresentasikan dengan menggunakan *projection layer* sehingga menghasilkan multi vektor. Multi vektor ini kemudian **dirata-ratakan** menjadi suatu vektor dan masuk ke *output layer*. Fungsi aktivasi *softmax* digunakan pada *output layer* untuk menghasilkan prediksi. Untuk optimalisasi model dengan banyak kosakata digunakan *hierarchical softmax* dan output yang direpresentasikan dengan *huffman tree* yang dibentuk berdasarkan frekuensi kata.



Gambar 2.9.1 Arsitektur CBOW
Sumber: (Mikolov, Chen, dkk., 2013, hlm. 5)

Setelah proses pelatihan model, vektor yang dihasilkan dari *layer projection* memiliki nilai yang merepresentasikan setiap kata, sehingga vektor tersebut dapat dioperasikan. Misalnya untuk mencari kata yang mirip dengan *small* dalam arti yang sama dengan *biggest* mirip dengan *big*, dapat dihitung dengan cara X = vektor("*biggest*") – vektor("*big"*) + vektor("*small"*). *Embedding* suatu kata yang dihasilkan dicari kemiripannya dengan *embedding* kata lain di dalam vektor menggunakan *cosine distance* (1-*cosine similarity*). Adapun *cosine similarity* dihitung dengan persamaan **(23)** berikut.

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}} \quad \textbf{(23)}$$

Bila model telah dilatih maka mungkin menemukan jawaban yang benar, yakni *cosine distance* yang paling kecil, menggunakan metode ini. Untuk mengevaluasi kualitas vektor, digunakan suatu set penalaran analogis yang

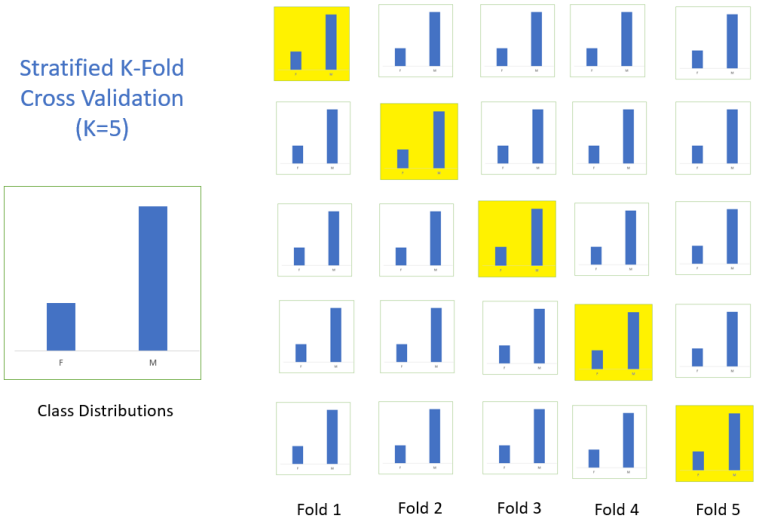
berisi kata dan frasa. Pasangan analogis ini dibentuk dengan cara mengambil frasa yang sering muncul lalu memberikan kata representasinya dari frasa tersebut. Misalnya, "Montreal":"Montreal Canadiens", "Toronto":"Daun Maple Toronto". Jawaban dianggap benar jika representasi terdekat dari vektor("Montreal Canadiens") - vektor("Montreal") + vektor("Toronto") adalah vektor("Toronto Maple Leafs") (Mikolov, Chen, dkk., 2013; Mikolov, Sutskever, dkk., 2013).

Evaluasi dapat juga dilakukan secara intrinsik yakni visualisasi. *Clustering* dapat memberikan visualisasi dengan mengambil **n** kata yang sering muncul dan memvisualisasikannya. Algoritma k-means digunakan untuk visualisasi pengelompokkan karena mempunyai skor *purity* yang tinggi (Zhai dkk., 2016). Dimensi vektor yang tinggi perlu dikurangi menjadi dua dengan menggunakan PCA (*Principal Component Analysis*) atau t-SNE (*t-distributed Stochastic Neighbor Embeddings*) (Liu dkk., 2018). Perbedaannya adalah PCA mengurangi dimensi secara linear dengan memetakan data berdimensi tinggi secara linear ke dimensi yang lebih rendah sambil memaksimalkan variasi dari data. Sedangkan, t-SNE menghitung kemiripan pada dimensi tinggi dan dimensi rendah lalu menggunakan metode optimalisasi, misalnya *gradient descent*, untuk meminimalisir perbedaan antara kedua dimensi tersebut (Winastwan, 2020).

2.10 Stratified K-Fold Cross-Validation

Stratified K-Fold Cross-Validation (SKF) adalah suatu metode yang mengembalikan potongan data (*folds*) sebanyak **k** (**n_splits**) dan

memisahkannya menjadi set latih/uji untuk setiap potongan. Potongan tersebut dibuat dengan mempertahankan persentase sampel untuk setiap kelasnya pada masing-masing set (Scikit-Learn, 2021). Pada **Gambar 2.10.1** berikut ini adalah proses SKF untuk membentuk data latih/uji (Sunil, 2018).



Gambar 2.10.1 Ilustrasi Stratified K-Fold Cross Validation
Sumber: (Sunil, 2018, bag. 4)

2.11 Konversi nilai ke rentang baru

Memetakan ulang nomor dari satu rentang ke rentang lainnya dapat dilakukan dengan menggunakan fungsi berikut (Arduino, 2022).

```
long map(long x, long in_min, long in_max, long out_min, long out_max) {
  return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
}
```

Gambar 2.11.1 Fungsi untuk konversi nilai ke rentang lain