Teknik *Distributed* Naive Bayes untuk Analisis Sentimen Ulasan Pelanggan Amazon

Distributed Naive Bayes *Technique For Sentiment Analysis Of Amazon Customer Reviews*

Baiq Wilda Al Aluf[1], Ari Hernawan, S.Kom., M.Sc.[1], Gibran Satya Nugraha, S.Kom., M.Eng.[2]
[1]Dept Informatics Engineering, Mataram University
Jl. Majapahit 62, Mataram, Lombok NTB, INDONESIA
Email: baiqaluf88@gmail.com, arihernawan@unram.ac.id, gibransn@ unram.ac.id

Big data analysis is an essential tool and technique for understanding large datasets, with the condition that the algorithms used can handle large amounts of data. To address this, Apache Spark is employed as a parallel computing framework that can operate across multiple clusters. This research utilizes the* Naive Bayes *classification method to analyze positive and negative customer reviews on Amazon's website. The objective of this study is to measure the performance of distributed* Naive Bayes *and observe the running time required for text classification in sentiment analysis of user reviews on the Amazon website using Apache Spark. Evaluating the running time is used to assess the efficiency of data processing with Apache Spark. The results of implementing* Naive Bayes *on Spark using 4 node workers and 3 datasets demonstrate success, as the model achieves good accuracy, namely 82.31% for a dataset size of 500 MB, 82.42% for 1000 MB, and 82.23% for 1500 MB, all with efficient processing times using 4 worker nodes.. Significant reductions in running time are observed when varying the number of node workers used. However, for a broader understanding of system performance in more complex scenarios, further research is required, such as exploring the use of more than 4 node workers. Thus, this research serves as a promising initial step in expanding our understanding of system performance in more complex conditions.

Key words*: *Big data,* Apache Spark, Naive Bayes, *Distributed Computing, Customer Review Classification of Amazon

I. INTRODUCTION

Dalam era perkembangan teknologi saat ini, *big data* telah menjadi fenomena yang sangat penting dengan pertumbuhan yang cepat dalam ukuran dan kompleksitasnya. Setiap hari, volume data yang dihasilkan melebihi kapasitas pemrosesan komputasi konvensional, dengan tingkat pertumbuhan rata-rata sekitar 60% setiap tahun. *Big data* mencakup berbagai jenis data, termasuk teks, gambar, video, suara, sensor, dan data transaksi. Pengelolaan dan pemrosesan *big data* bukanlah tugas yang mudah dan memerlukan pendekatan khusus [1].

Untuk menghadapi tantangan pemrosesan *big data*, diterapkanlah Sistem Komputasi Terdistribusi yang terdiri dari beberapa unit komputer atau *node worker* yang bekerja bersama-sama untuk memproses data secara terdistribusi [2]. Salah satu kerangka kerja yang dapat efisien mengolah *big data* adalah Apache Spark. Apache Spark beroperasi dengan menggunakan konsep RDD (Resilient Distributed Dataset) yang memungkinkan pemrosesan data secara paralel di banyak *cluster* dengan cepat dan efisien [3]. Melalui API PySpark pada Apache Spark, dapat dilakukan transformasi, pemrosesan, dan analisis data dalam skala besar secara terdistribusi pada kluster Spark

Salah satu *framework* yang dapat mengolah *big data* adalah Apache Spark [4]. Apache Spark dapat digunakan untuk berbagai aplikasi pemrosesan data, termasuk machine learning. Apache Spark bekerja pada konsep dasar yang disebut dengan RDD (Resilient Distributed Dataset), yaitu kumpulan data terdistribusi yang dapat diproses secara parallel di banyak *cluster*. RDD memungkinkan Apache Spark untuk melakukan pemrosesan data dengan cepat dan efisien [5]. Selain itu Apache Spark menyediakan API PySpark yang mendukung konsep *Resilient Distributed Datasets* (RDD) dalam bahasa pemrograman Apache Spark dan Python yang dapat digunakan untuk mengoperasikan data terdistribusi [1]. Apache Spark juga memiliki komponen *library open-source* yaitu *Machine Learning library* (MLlib) yang merupakan sebuah *library* yang memungkinkan Apache Spark untuk melakukan fungsi-fungsi umum dari *machine learning* seperti melakukan klasifikasi [2].

Penelitian ini menggunakan analisis *big data* untuk analisis sentimen guna memahami perilaku pengguna dengan melihat *review*/ulasan pada *Amazon* yang terkumpul pada dataset Amazon Reviews. Ulasan pelanggan di Amazon memberikan masukan berharga bagi perusahaan dan calon pembeli. Ulasan ini membantu calon pembeli dalam membuat keputusan berdasarkan pengalaman pengguna lain, sementara bagi Amazon, ulasan tersebut menjadi umpan balik untuk meningkatkan produk dan strategi pemasaran [3]. Melalui analisis sentimen, Amazon dapat mengkategorikan ulasan pengguna sebagai positif atau negatif, sehingga dapat meningkatkan kualitas produk dan mengoptimalkan strategi pemasaran. Analisis ini memiliki pengaruh yang signifikan terhadap reputasi perusahaan dan kemampuannya dalam memasarkan produk [4].

Dalam melakukan analisis sentimen ulasan pengguna pada *Amazon* digunakan metode klasifikasi dengan menggunakan Naive Bayes untuk *distributed computing* dengan menggunakan Apache Spark terutama dari sisi kecepatan secara spesifik pada proses analisis sentimen pengguna *Amazon* [5]. Kinerja algoritma Naive Bayes dapat dilihat dari berapala lama waktu yang dibutuhkan saat *running time* pada proses klasifikasi teks untuk analisis sentimen ulasan pengguna pada Amazon dilaksanakan. Sehingga pada penelitian ini akan menggunakan metode Naive Bayes dalam pengklasifikasian datanya dan akan dibantu dengan penerapan *Distributed Computing System* agar dapat memaksimalkan metode Naive Bayes dalam mengolah data yang sangat besar [6]. Pengujian dilakukan untuk melihat kinerja penggunaan Naive Bayes terdistriusi menggunakan Apache Spark pada proses klasifikasi dengan melakukan perbandingan hasil *running time* pada setiap penggunaan jumlah *node worker* yang berbeda. Hal ini dilakukan untuk mengetahui jumlah *node worker* yang paling efektif digunakan pada proses klasifikasi. Evaluasi *running time* yang didapatkan sangat penting untuk mengukur seberapa cepat proses pengolahan data dilakukan dengan menggunakan Apache Spark pada berbagai ukuran *cluster*

II. TINJAUAN PUSTAKA

A. Penelitian Terkait

Pertama, penelitian dari T. Tekdogan dan A. Cakmak dengan judul "Benchmarking Apache Spark and Hadoop MapReduce on *Big data* Classification [4]. Pada penelitian ini membahas tentang mengidentifikasi kelebihan dan kekurangan Apache Spark dan Hadoop MapReduce dalam *big data* yang diilustrasikan melalui tugas klasifikasi pada 4 (empat) dataset dengan menggunakan algoritma Naive Bayes Classiffier. Hasilnya menunjukkan bahwa Spark sekitar lima kali lebih cepat daripada Hadoop dalam hal waktu pelaksanaan fase pelatihan.

Kedua, penelitian dari Y. Samadi, M. Zbakh, and C. Tadonki dengan judul "Performance comparison between hadoop and spark frameworks using HiBench benchmarks [5]. Penelitian ini membahas perbandingan kinerja antara dua kerangka *Big data* populer yang diterapkan pada mesin virtual yaitu Apache Spark pada mode terdistribusi semu dan menerapkan Hadoop pada lingkungan komputasi awan. Hasil dari analisis yang dilakukan adalah bahwa ada beberapa faktor dapat menimbulkan perbedaan kinerja yang signifikan Spark memiliki lebih banyak pengoptimalan, seperti jumlah akses disk per detik, pemanfaatan bandwidth memori, dan laju IPC daripada Hadoop, sehingga memberikan kinerja yang lebih baik.

Ketiga, penelitian dari C. Wibawa, S. Wirawan, M. Mustikasari, dan D. T. Anggraeni, "Komparasi Kecepatan Hadoop Mapreduce Dan Apache Spark Dalam Mengolah Data Teks [1]. Penelitian ini membahas tentang membandingkan kecepatan dan performa antara Hadoop MapReduce dan Apache Spark dalam memproses data teks yang besar. Hasil dari studi menunjukkan bahwa Apache Spark memiliki performa yang lebih baik dan lebih cepat dalam memproses data teks yang besar dibandingkan dengan Hadoop MapReduce. Apache Spark mampu memproses data teks lebih cepat dengan waktu pemrosesan yang lebih singkat dan efisiensi yang lebih baik. Studi ini menunjukkan bahwa Apache Spark dapat menjadi pilihan yang lebih baik untuk memproses data teks yang besar dengan kecepatan yang lebih tinggi dan performa yang lebih baik.

Keempat, penelitian dari B. Liu, E. Blasch, Y. Chen, D. Shen, dan G. Chen, dengan judul "Scalable Sentiment Classification for *Big data* Analysis Using Naïve Bayes Classifier" [12]. Penelitian ini membahas tentang pengujian performa dari algoritma klasifikasi sentimen dengan Naive Bayes pada beberapa dataset yang besar, dan membandingkan performa dari algoritma tersebut pada platform Apache Spark dengan implementasi serial dari algoritma tersebut. Hasil dari studi menunjukkan bahwa implementasi pada platform Apache Spark dapat meningkatkan performa dan skalabilitas dari algoritma klasifikasi sentimen pada dataset yang besar, dengan waktu pemrosesan yang lebih cepat dibandingkan dengan implementasi serial.

Kelima, penelitian dari Xiaofang Wang, Lan Luo, Zou Qianyin, and etc. dengan judul "Constructing Naive Bayesian Classification Model By Spark For *Big data*" [13]. Penelitian ini membahas tentang penggunaan model klasifikasi Naive Bayes pada data yang besar dengan menggunakan teknologi Apache Spark. Dilakukan pembangunan model klasifikasi Naive Bayes untuk *Big data* menggunakan teknologi Spark. Konstruksi model klasifikasi Naive Bayes dilakukan dengan memanfaatkan kemampuan Spark dalam melakukan proses distribusi data. Pengujian dilakukan pada dataset yang cukup besar dengan menggunakan tiga jenis klasifikasi yaitu Naive Bayes pada Spark, Naive Bayes pada platform standar (Tanpa Spark), dan Metode Support Vector Machine (SVM) dengan Spark. Hasil pengujian menunjukkan bahwa metode klasifikasi Naive Bayes pada Spark memberikan akurasi yang lebih baik dan waktu proses yang lebih singkat dibandingkan dengan metode klasifikasi Naive Bayes pada platform standar dan metode SVM pada Spark dengan akurasi prediksi model Naive Bayes Spark MLlib sekitar 82,93% .

Keenam, penelitian dari C. Darujati and A. Bimo Gumelar dengan judul "Pemanfaatan Teknik Supervised Untuk Klasifikasi Teks Bahasa Indonesia" [14]. Penelitian ini membahas tentang pengembangan dan pemanfaatan aplikasi untuk mengklasifikasi teks dalam bahasa Indonesia dengan bantuan metode Naive Bayes yang terpandu (supervised). Dalam penelitian ini, penulis menerapkan metode Naive Bayes Classifier menggunakan TF-IDF untuk melakukan klasifikasi berita. Hasil penelitian menunjukkan bahwa kombinasi TF-IDF dan Naive Bayes mampu mencapai tingkat akurasi hingga 87%. Hal ini menunjukkan bahwa pendekatan tersebut efektif dalam mengklasifikasikan teks bahasa Indonesia dengan tingkat keberhasilan yang tinggi. Penelitian ini memberikan bukti bahwa metode TF-IDF dan Naive Bayes

dapat digunakan sebagai alat yang berguna dalam analisis teks bahasa Indonesia dan dapat memberikan hasil yang memuaskan dalam klasifikasi Penelitian ini juga menunjukkan bahwa klasifikasi dapat berjalan dengan baik ketika menggunakan data latih yang berjumlah besar, yakni sebanyak 100 artikel. Selain itu, hasil eksperimen juga menunjukkan bahwa klasifikasi masih memberikan hasil yang memadai ketika menggunakan lebih dari 150 dokumen sebagai data latih.

Berdasarkan dari tinjauan pustaka penelitian yang akan dilakukan adalah menganalisis kinerja algoritma Naive Bayes dengan menggunakan sistem terdistribusi Apache Spark menggunakan beberapa jumlah *node worker* pada Spark *cluster* untuk analisis sentimen pengguna Amazon.

B. Teori Penunjang

Apache Spark merupakan sebuah framework untuk melakukan proses analisis *big data* open source, cepat, dan disimpan dalam memori. Spark memungkikan pengguna untuk melakukan proses analisis *big data* open source, cepat, dan disimpan dalam memori [15].

Big data merupakan kumpulan data yang beukuran sangat besar dan kompleks, sehingga untuk diproses tidak memungkinkan untuk menggunakan perangkat pengelola *database* konvensional ataupun aplikasi pemroses data lainnya [7]

Big data analitycs merupakan alat dan teknik analisis yang akan sangat membantu dalam memahami *big data* dengan syarat algoritma yang digunakan harus mampu bekerja dengan jumlah besar pada kondisi real-time dan pada data yang berbeda-beda [17]. Tujuan utama dari *big data analytics* adalah untuk mendapatkan big value/informasi yang berharga, sehingga dapat dimanfaatkan dalam berbagai bidang yang mengarah pada pengambilan keputusan yang lebih baik.

Analisis big data dapat dibedakan dari arsitektur tradisional data *processing* di sejumlah dimensi:

1. *Volume*, *big data* dapat memproses data secara real-time atau mendekati real-time, sedangkan sistem tradisional cenderung bekerja secara batch, memproses data dalam interval tertentu.
2. *Velocity*, *big data* juga melibatkan data yang dihasilkan dengan kecepatan tinggi atau *real-time*. Data ini terus-menerus mengalir dan harus segera diolah untuk mendapatkan wawasan yang berguna.
3. *Variety*, *big data* mencakup berbagai jenis data, termasuk teks, gambar, audio, video, data sensor, data transaksi, data sosial media, dan banyak lagi. Keanekaragaman jenis data ini menambah kompleksitas dan tantangan dalam pengolahan dan analisis.
4. *Veracity*, *big data* seringkali memiliki struktur yang tidak teratur atau tidak terstruktur. Data ini mungkin tidak mematuhi skema atau format yang terorganisir dengan baik seperti dalam basis data relasional tradisional.
5. *Value*, *value* dari *big data* terletak pada potensi informasi yang dapat diambil darinya. Data ini dapat memberikan wawasan dan pemahaman yang lebih mendalam tentang tren, pola, preferensi pengguna, dan peluang bisnis yang dapat digunakan untuk pengambilan keputusan yang lebih baik.

Pyspark merupakan modul Python untuk mengakses, memproses, dan menganalisis data dengan bantuan Apache Spark. Apache Spark menggunakan model pemrosesan *in-memory* dan menggunakan *cluster* komputasi untuk memproses data secara *parallel*, memungkinkan tugas pemrosesan data besar diselesaikan dengan cepat [1].

Analisis sentimen merupakan proses memahami, mengekstrak dan mengolah data tekstual secara otomatis untuk mendapatkan informasi sentimen yang terkandung dalam suatu kalimat opini. Analisis sentimen dilakukan untuk melihat pendapat atau kecenderungan opini terhadap sebuah masalah atau objek oleh seseorang, apakah cenderung berpandangan atau beropini negatif atau positif [8].

Preprocessing adalah serangkaian langkah yang dilakukan untuk mempersiapkan data mentah agar dapat diolah dan dianalisis dengan algoritma machine learning atau statistik. *Preprocessing* menjadi tahap awal dalam klasifikasi teks untuk mempersiapkan data teks sebelum digunakan pada proses lainnya. Pada tahap ini akan mengubah data teks menjadi bentuk yang lebih baik sehingga menghasilkan informasi teks dengan kualitas yang baik dan siap digunakan pada proses selanjutnya. Tahapan-tahapan tersebut dibagi menjadi *case folding*, *tokenizing*, *filtering* dan *stemming* [9].

Metode TF-IDF (*Term Frequency-Inverse Document Frequency*) merupakan sebuah teknik ekstraksi fitur untuk teks yang berfungsi untuk menentukan bobot kata-kata (*term*) dalam sebuah dokumen. Metode TF-IDF menggabungkan dua konsep utama, yaitu frekuensi kemunculan suatu kata dalam dokumen tertentu (TF) dan kebalikan frekuensi kemunculan kata tersebut dalam seluruh dokumen (IDF). Berikut umus TF (*Term Frequency*) menggambarkan seberapa sering suatu term (kata) muncul dalam suatu dokumen. Rumus TF dapat dinyatakan sebagai berikut:

$$TF\ (t,d) = \frac{jumlah\ kemunculan\ term\ t\ dalam\ dokumen\ d}{jumlah\ total\ term\ dalam\ doukum\ d} \quad (2\text{-}1)$$

Di mana:

- TF(t, d) adalah skor TF untuk term t dalam dokumen d.

Sedangkan IDF (*Inverse Document Frequency*) merupakan metrik yang digunakan dalam perhitungan TF-IDF untuk menentukan seberapa penting suatu term (kata) dalam koleksi dokumen secara keseluruhan. IDF dihitung dengan rumus sebagai berikut:

$$log\ \frac{\mathrm{n}}{\mathrm{df(t)}} \quad (2\text{-}2)$$

Di mana:

- IDF adalah nilai IDF dari kata kunci.
- N adalah jumlah total dokumen dalam koleksi.

- df (t) adalah jumlah dokumen yang mengandung kata kunci (term).

Naive Bayes *Classifier* adalah algoritma pembelajaran mesin yang digunakan untuk melakukan klasifikasi [10]. Algoritma ini didasarkan pada teorema Bayes, yang merupakan konsep dasar dalam probabilitas. Prinsip dasar dari algoritma ini adalah menghitung probabilitas setiap label kelas (misalnya, positif atau negatif) berdasarkan fitur-fitur *input* yang ada [5]. Berikut adalah rumus dasar untuk Naive Bayes adalah sebagai berikut:

$$P(C|X) = P(X|C) * P(C)/P(X) \quad (2\text{-}1)$$

Di mana:

- P(C|X) adalah probabilitas kelas C, jika diberikan X (data baru)
- P(X|C) adalah probabilitas nilai atribut X, jika diberikan kelas C
- P(C) adalah probabilitas dari kelas C
- P(X) adalah probabilitas dari nilai atribut X

Akurasi merupakan anuran yang menunjukkan seberapa baik sebuah model atau sistem dapat mengenali sentimen dengan benar. Akurasi dihitung dengan membagi jumlah sentimen yang terklasifikasi dengan benar oleh model dengan jumlah total data yang diuji [10]. Dalam perhitungan akurasi, akan membandingkan sentimen yang telah diklasifikasikan oleh model dengan sentimen yang sebenarnya atau label yang benar pada data uji.

$$akurasi = \frac{(TP+TN)}{(TP+FP+FN+TN)} \times 100\% \quad (2\text{-}2)$$

Di mana:
TP = *True Positive*
TN = *True Negative*
FP = *False Positive*
FN = *False Negative*

III. METODE PENELITIAN

A. Alat dan Bahan

Alat dan bahan yang dibutuhkan pada penelitian ini berupa *software*, *hardware* serta data dan informasi pendukung selama dilakukannya penelitian.

A.1. Alat

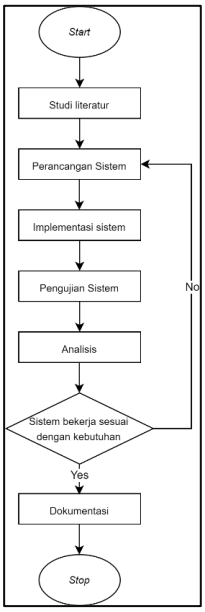
1. Laptop dengan sistem operasi windows 10 digunakan untuk dokumentasi penelitian.
2. 5 *Virtual Private Server* (VPS) yang digunakan untuk keperluan spark *cluster*.
3. Pyspark API digunakan untuk pemrosesan data terdistribusi menggunakan Apache Spark.
4. *Jupyter Notebook* digunakan untuk menjalankan pemrogram.
5. Spark *Framework* digunakan untuk pemrosesan *big data*.

A.2. Bahan

Bahan yang digunakan pada penelitian ini adalah studi literatur yang diambil adri buku, jurnal dan penelitian yang telah dilakukan sebelumnya. Bahan lainnya yang digunakan dalam penilitian ini adalah dataset Amazon Reviews. Di mana dataset Amazon Reviews berisikan ulasan-ulasan dari pelanggan atau pengguna produk yang dibeli atau digunakan melalui platform Amazon yang bersumber dari Kaggle dengan ukuran 1.76 GB. [11]. Di mana isi *reviews* pada dataset ini menggunakan Bahasa asing . Dataset ini terdapat dua kelas yaitu kelas 1 untuk kelas negatif dan kelas 2 untuk kelas positif. Untuk data *testing*-nya berjumlah 399.99 *record* dengan 200.000 untuk kelas negatif dan 199.999 untuk kelas positif. Untuk kelas *training*-nya berjumlah 3.599.999 *record* dengan 1.800.000 untuk kelas negatif dan 1.799.999 untuk kelas positif.

B. Alur Penelitian

Penelitian ini dilakukan melalui beberapa tahapan yaitu studi literatur dan persiapan alat dan bahan, perancangan sistem, implementasi sistem, pengujian sistem, dan dokumentasi. Alur langkah penelitian yang dilakukan dapat dilihat pada Gambar 3.1.



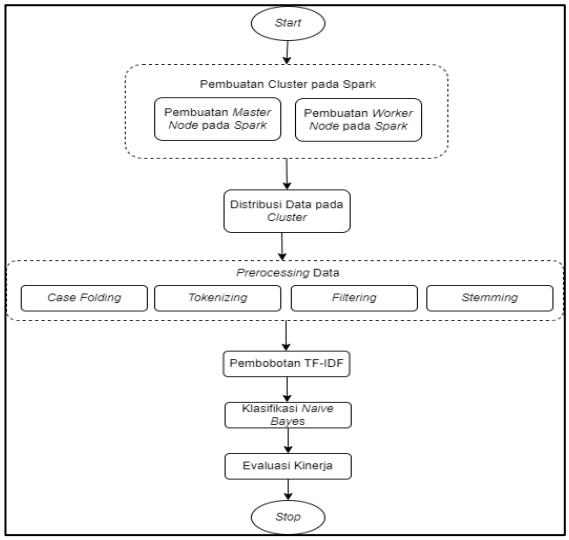
Gambar. 1. Alur Penelitian

1. Studi literatur, pada tahap ini dilakukan analisis kebutuhan tehadap pengembangan sistem dalam penelitian yang dilakukan dengan mencari referensi literatur yang berkaitan dengan topik penelitian.
2. Perancangan sistem, pada tahap ini dilakukan perancangan sistem secara terkonsep seperti bagaimana alur kerja dari sistem.
3. Implementasi sistem, pada tahap ini dilakukan penerapan sistem sesuai dengan konsep yang telah dirancang sebelumnya. Diawali dengan pembuatan *cluster* Spark, analisis sentimen menggunakan Naive Bayes, proses pengujian, dan evaluasi.
4. Pengujian sistem, pada tahan ini dilakukan dua pengujian, yang pertama dilakukan pengujian model yang telah dibuat. Pengujian kedua dilakukan untuk mengetahui berapa lama *running time* yang dibutuhkan pada beberapa jumlah worker saat proses training

dilakukan. Di mana, pada pengujian ini digunakan 4 *node worker* pada *cluster* Spark.

5. Analisis, pada tahap ini dilakukan analisis terhadap hasil yang didapatkan dari proses pengujian.
6. Dokumentasi, pada tahap ini dilakukan dokumentasi berupa laporan dari hasil yang dilakukan berupa proses analisis kinerja algoritma distributed Naive Bayes untuk mengklasifikasikan ulasan negatif dan positif pengguna Amazon.

C. Perancangan Sistem



Gambar. 2. Perancangan Sistem

1. Pembuatan *Master Node* pada Spark, yaitu pembuatan *master node* dengan menginstal spark *master node* untuk mengkoordinasi tugas-tugas dalam *cluster*.
2. Pembuatan *Worker Node* pada Spark, yaitu penginstalan *worker*. Di mana dilakukan penginstalan 4 *worker node* pada Spark yang akan terhubung ke Spark *Master Node* yang telah dibuat sebelumnya.
3. Distribusi Data pada *cluster*, yaitu proses untuk mendistribusikan data pada *cluster* Spark.
4. *Case Folding*, yaitu proses pengkonversian karakter pada dataset dalam sebuah teks ke bentuk yang sama dari huruf besar menjadi huruf kecil dan menghilangkan tanda baca pada kalimat.
5. *Tokenizing*, yaitu penghilangan karakter-karakter tertentu pada dataset, seperti setiap kata akan dipisahkan berdasarkan spasi yang ditemukan
6. *Filtering*, yaitu pembuangan kata-kata yang tidak penting pada dataset.
7. *Stemming*, yaitu mengubah kata berimbuhan menjadi kata dasar pada dataset
8. Pembobotan TF-IDF, yaitu proses menghitung bobot yang bertujuan untuk mengidentifikasi seberapa besar pengaruh suatu kata dalam suatu teks.
9. Klasifikasi Naive Bayes, yaitu pengklasifikasian yang akan menggunakan metode Naive Bayes untuk mendapatkan hasil klasifkasi ulasan negatif dan positif dari pengguna Amazon.
10. Pengujian, yaitu proses pengujian untuk menentukan sentimenn terhadap ulasan Amazon oleh pengguna dari data yang telah diklasifikasikan sebelumnya. Pengujian juga dilakukan dengan melihat *running time* dari proses klasifikasi dengan menggunkan jumlah *node* yang berbeda.
11. Evaluasi, yaitu sebagai tolak ukur efektivitas klasifikasi yang dihasilkan pada pengujian sebelumnya. Selain itu, evaluasi *running time* juga dilakukan pada setiap jumlah *node* yang berbeda pada *cluster* yang digunakan untuk pemrosesan machine learning.

D. Metode

Penelitian ini menggunakan Naive Bayes Classifier sebagai metode pengklasifikasian dataset Amazon Reviews untuk mendapatkan analisis sentimen berupa kelas positif atau negatif. Naive Bayes Classifier merupakan sebuah metode klasifikasi yang berakar pada teorema Bayes, yang digunakan untuk memprediksi kemungkinan kejadian di masa depan berdasarkan pengalaman sebelumnya. Berikut contoh perhitungan probabbilitas Naive Bayes.

Pada proses pengklasifikasian diperlukan hasil dari data yang sudah diolah dari proses sebelumnya yaitu hasil dari proses *preprocessing*. Selanjutnya data yang telah di *preprocessing* akan diklasifikasikan menggunakan metode Naive Bayes *Classifier* dengan penggunaan beberapa *worker* dalam sebuah *cluster*, data training akan dibagi menjadi beberapa subset. Setiap subset tersebut akan diproses secara paralel oleh beberapa *worker* yang berbeda dalam *cluster*. Dengan menggunakan beberapa *worker* dalam *cluster*, penggunaan Naive Bayes *Classifier* dapat dipercepat secara signifikan untuk pemrosesan dataset yang besar. Pendekatan ini memanfaatkan kekuatan pemrosesan paralel dan distribusi dalam *cluster*

E. Pengujian

Proses pengujian yang dilakukan adalah melakukan perbandingan penggunaan beberapa jumlah *worker* pada proses klasifikasi analisis sentimen pengguna *Amazon* menggunakan Apache Spark.

E.1. Metrik Pengujian

1. Akurasi, metrik pegujian akurasi digunakan untuk mengukur kinerja Naive Bayes dalam mengklasifikasikan sentimen dengan benar.
2. *Running time*, metrik pengujian *running time* digunakan untuk mengukur waktu yang diperlukan untuk melatih model Naive Bayes dalam melatih dataset lalu membandingkan kecepatan *running time* pada setiap jumlah *worker* yang berbeda.

E.2. Alur Pengujian

1. Membuat konfigurasi *cluster* Apache Spark yang berjumlah 4 *node worker* dan harus terhubung dengan Spark Master serta memiliki akses ke data yang sama.
2. Meng-*input* data yang dilakukan pada masing-masing *node worker* yang aktif.

3. Melakukan pra-pemrosesan data pada masing-masing *node worker*.
4. Melakukan pembobotan TF-IDF.
5. Melakukan klasifikasi Naive Bayes yang dijalankan pada masing-masing *node worker*.
6. Melakukan evaluasi dari setiap *node worker* dan menggabungkan seluruh hasil evaluasi untuk mendapatkan hasil keseluruhan.
7. Melakukan analisis terhadap waktu pelatihan dan evaluasi model.
8. Melakukan penambahan jumlah penggunaan *node worker* dan mengulangi tahapan sebelumnya hingga penggunaan *node worker* berjumlah 4.
9. Membandingkan kinerja Naive Bayes dari setiap pengujian yang telah dilakukan. Penelitian dianggap berhasil jika semua tahapan pengujian selesai dilaksanakan.

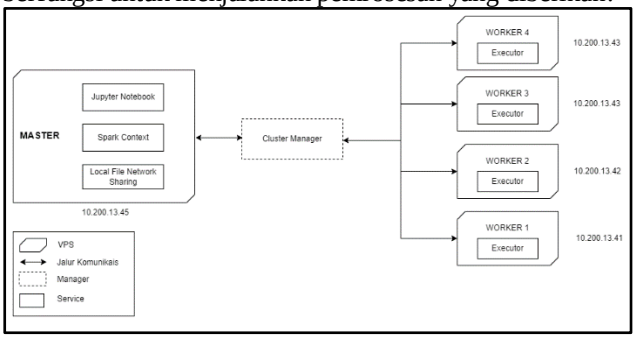
F. Cara Analisis

Analisis yang dilakukan yaitu membandingkan hasil kinerja model Naive Bayes yang terdiri dari waktu pelatihan, evaluasi model, dan penggunaan sumber daya saat menggunakan jumlah *node worker* yang berbeda. Dari analisis tersebut dapat dibuat kesimpulan terkait berapa jumlah *node woker* yang memiliki kinerja optimal dan memberikan hasil evaluasi yang baik dalam klasifikasi analisis sentimen ulasan pengguna *Amazon* menggunakan Naive Bayes terdistribusi pada Spark.

IV. HASIL DAN PEMBAHASAN

A. Implementasi SVM pada Spark

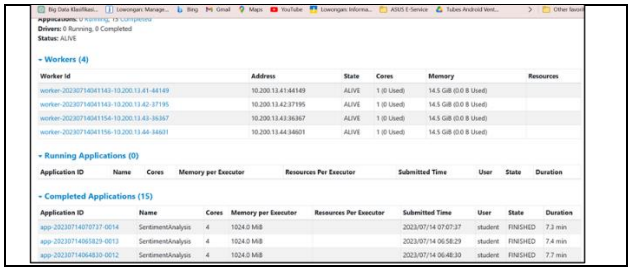
Pada penelitian ini, *cluster* Spark yang dirancang terdiri dari satu *node master* dan 4 *node worker*. Peran utama dari *node master* adalah sebagai pengelola sumber daya dan tugas-tugas di dalam *cluster*. Sementara itu, *node worker* berfungsi untuk menjalankan pemrosesan yang diberikan.



Gambar. 3. Topologi *Cluster* Spark

Pada Gambar 3 terdapat topologi *cluster* Spark yang digunakan untuk melakukan komputasi metode Naive Bayes secara terdistribusi. Pada *master node* didalamnya terdapat *Jupyter Notebook* untuk menjalankan kode pemrograman. Di dalam Spark *cluster*, *Jupyter Notebook* dapat berinteraksi dengan Spark *cluster* menggunakan PySpark. PySpark adalah API Python untuk Spark yang memungkinkan akses dan operasi Spark melalui bahasa pemrograman Python. Pada *Jupyter Notebook* terlebih dahulu dilakukan pembuatan sesi pada Spark yang digunakan untuk Spark berinteraksi langsung dengan Spark *cluster*. Jupyter Notebook dapat dijalankan di port 8888 pada *node master*. *Node master* dapat berkomunikasi dengan *node worker* melalui TCP/IP seperti yang terdapat pada gambar di atas.

Pada pengujian ini dilakukan menggunakan VPS (Virtual Private Server), yaitu bentuk layanan hosting di mana server fisik dibagi menjadi beberapa server virtual yang berfungsi secara independen. Setiap VPS memiliki sistem operasi dan sumber daya yang terisolasi, seperti CPU, RAM, dan ruang penyimpanan, sehingga memberikan kesan memiliki server fisik sendiri. Di mana pada VPS yang digunakan terdapat 1 master dan 4 *node worker*



Gambar. 4. *Interface* Spark Master

Pada Gambar 4 terlihat *user interface* dari Spark Master yang telah dibuat sebelumnya. *User interface* ini menampilkan informasi rekaman dari Spark master berupa semua aktivitas yang terjadi pada setiap *node worker* dalam *cluster* saat sebuah proses sistem dijalankan, mulai menampilkan *node worker* yang aktif dari waktu awal eksekusi hingga waktu berakhirnya proses tersebut . User interface Spark master ini dapat diakses melalui port 8080. Dalam penelitian ini, hasil *running time* dari proses klasifikasi yang dilakukan oleh sistem dapat dilihat melalui interface spark *cluster*.

Dalam proses pelatihan Naive Bayes, data *training* akan didistribusikan ke 4 *node worker* untuk diproses secara paralel. Setiap *node worker* akan menjalankan algoritma pelatihan Naive Bayes pada bagian data yang diterima. Hasil dari proses tersebut akan dikumpulkan oleh master *node* untuk digabungkan menjadi satu model Naive Bayes yang lengkap. Data yang akan diproses oleh Naive Bayes di dalam Spark *Cluster* disimpan dalam penyimpanan lokal di setiap *node* dalam *cluster*. Selanjutnya, Apache Spark menggunakan *library machine learning* bernama Spark MLlib yang menyediakan implementasi Naive Bayes untuk melatih model Naive Bayes dan menerapkan *Naive Baye* yang dapat dijalankan di dalam Spark *cluster*. Penulisan kode dilakukan pada *Jupyter Notebook*, yang dapat berinteraksi dengan *cluster* Spark menggunakan PySpark, yaitu Python API untuk Spark.

B. Distribusi Data

Data yang akan diproses dalam Spark *Cluster* didistribusikan dan disimpan secara lokal di setiap *node*

dalam *cluster*. Tujuannya adalah untuk memastikan bahwa setiap *node* memiliki akses untuk membaca data yang akan diproses oleh Spark. Data tersebut terbagi menjadi tiga ukuran yaitu 500 MB, 1000 MB, dan 1500MB. Dalam penelitian ini, data dengan ukuran 500 MB disimpan dalam file 500test.csv dan 500train.csv, data dengan ukuran 1000 MB disimpan dalam file testnb1.csv dan nb1.csv, dan data dengan ukuran 1500 MB disimpan dalam file testnb2.csv dan nb2.csv. Ketika melakukan pengujian dengan data yang berbeda, program akan membaca file yang sesuai dengan data yang akan diuji.

C. Preprocessing Data

Pada tahap pre processing data akan dilakukan pembersihan data terlebih dahulu dengan menggunakan case folding, filtering, stemming dan toknizing. hal ini dilakukan untuk memperiapkan data debelum diterapkan pada model atau algoritma tertentu.

D. TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency) adalah sebuah metode yang digunakan dalam pemrosesan teks dan pengambilan informasi untuk mengukur tingkat pentingnya suatu kata dalam suatu dokumen relatif terhadap keseluruhan koleksi dokumen. Metode ini digunakan untuk memberikan bobot pada kata-kata dalam dokumen berdasarkan frekuensi kemunculan kata tersebut dalam dokumen (TF) dan frekuensi kemunculan kata tersebut dalam seluruh koleksi dokumen (IDF).

E. Hasil Pengujian

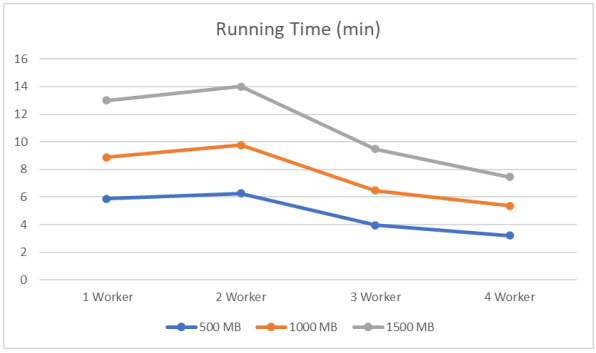
Pada pengujian ini, dilakukan variasi jumlah *node worker* dalam penggunaan Spark *cluster* yang dilakukan secara berurutan dari 1 *node worker* sampai 4 *node worker*. Selain itu, pada pengujian ini juga menggunakan dataset dengan ukuran yang berbeda, yaitu 500 MB, 1000 MB, dan 1.500 MB untuk melakukan pengujian. Berikut adalah tabel hasil *running time* dan akurasi yang dibutuhkan untuk analisis sentimen ulasan pelanggan Amazon menggunakan Spark dengan vairiasi jumlah *node worker*.

Ukuran Dataset	*Running time* (min)			
	1 Worker	2 Worker	3 Worker	4 Worker
500 MB	5.87	6.27	3.97	3.2
1000 MB	8.87	9.77	6.47	5.37
1500 MB	13	14	9.47	7.47

TABLE I. HASIL PENGUJIAN *RUNNING TIME*

Pada tabel I menampilkan evaluasi *running* pengujian oleh sistem untuk analisis sentimen ulasan pelanggan Amazon . pengujian ini menggunakan metode Naive Bayes terdistribusi pada Apache Spark dengan variasi jumlah *node worker*. Evaluasi ini didapatkan dari 3 kali pengujian yang dilakukan menggunakan Naive Bayes pada berbagai jumlah *node worker* dengan 3 ukuran dataset yang berbeda, yaitu 500 MB, 1000 MB dan 1500 MB.

Berdasarkan tabel di atas menunjukkan dengan menambah jumlah *node worker* dapat menghasilkan peningkatan *running time* secara signifikan, hal ini dapat dilihat oleh perbedaan hasil *running time* yang didapatkan menggunakan jumlah *node worker* yang berbeda pada tabel di atas. Di mana *node worker* yang menghasilkan *running time* paling optimal adalah penggunaan dengan 4 *node worker*. Hasil ini didapatkan karena dengan menggunakan 4 *node worker*, kapasitas pemrosesan sistem meningkat dan alokasi sumber daya yang tersedia dapat dioptimalkan dengan baik. Hal ini dapat menghasilkan *running time* yang lebih efisien dibandingkan dengan penggunaan jumlah *node worker* yang lebih sedikit.



Gambar. 1. Grafuk Hasil *Running Time*

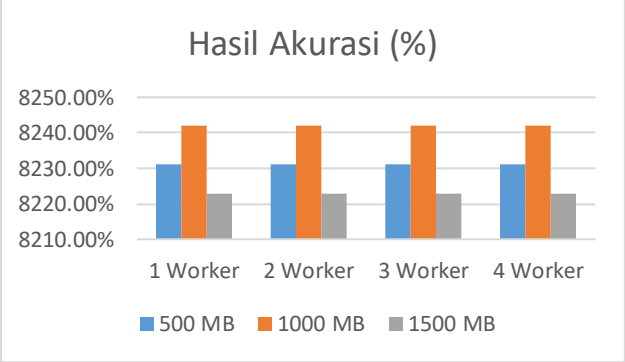
Pada Gambar 5 terdapat grafik yang memperlihat hasil evaluasi *running time* dari analisis sentimen ulasan pelanggan menggunakan algoritma Naive Bayes. Dapat dilihat dari grafik di atas hasil penelitian ini menunjukkan bahwa terjadi penurunan *running time* pada Spark ketika jumlah *worker node* ditambahkan. Di mana penggunaan 4 *node worker* memiliki hasil *running time* terendah dibandingkan dengan *node worker* lainnya. Namun, dapat dilihat pada penggunaan penggunaan 2 *node worker* memiliki sedikit perbedaan dan hasil rata-rata peningkatan *running time* dengan penggunaan 1 *node worker*. Hal ini menunjukkan bahwa kinerja Spark pada penggunaan 2 *node worker* tidak mengalami peningkatan yang signifikan. Sehingga dapat dikatakan penggunaan 2 *node worker* memiliki kinerja yang serupa dengan pengunaan 1 *node worker*, dengan hasil *running time* yang berada dalam rentang yang sama dengan *running time* pada penggunaan *node worker*.

Namun dapat dilihat juga pada penggunaan 3 dan 4 *worker node* terjadi penurunan *running time* yang menunjukkan peningkatan kinerja Spark. Dan penggunaan 4 *node worker*-lah yang menjadi jumlah *node worker* paling optimal pada implementasi Naive Bayes pada Spark dalam melakukan analisis sentimen ulasan pelanggan Amazon. Berdasarkan pengujian yang dilakukan penggunaan 4 *node worker* dianggap optimal karena hasil *running time* mengalami penurunan yang berarti kinerja spark mengalami peningkatan.

Ukuran Dataset	Akurasi (%)			
	1 Worker	2 Worker	3 Worker	4 Worker
500 MB	82.31	82.31	82.31	82.31
1000 MB	82.42	82.42	82.42	82.42
1500 MB	82.23	82.23	82.23	82.23

TABLE II. HASIL PENGUJIAN AKURASI

Pada tabel II menampilkan evaluasi akurasi yang dihasilkan oleh Naive Bayes pada 4 *node worker* dengan menggunakan 3 ukuran dataset yang berbeda, yaitu 500 MB, 1000 MB dan 1500 MB. Hasil analisis menggunakan algoritma Naive Bayes menunjukkan kinerja yang cukup baik, dengan akurasi mencapai 82.31% untuk dataset berukuran 500 MB, 82.42% untuk dataset berukuran 1000 MB dan 82.23% untuk dataset berukuran 1500 MB. Hasil akurasi pada variasi jumlah *node worker* ini menunjukkan bahwa tidak ada pengaruh apapun kepada hasil *running time* dari hasil akurasi yang dihasilkan. Hal ini dapat terjadi karena Spark memiliki kemampuan yang baik dalam mengelola peningkatan jumlah *node worker*.



Gambar. 2. Grafuk Hasil *Running Time*

Pada Gambar 2 menampilkan grafik dari hasil akurasi analisis sentimen ulasan pelanggan Amazon dengan menggunakan Naive Bayes. Dapat dilihat dari garfik di atas bahwa hasil akurasi dari berbagai jumlah *node worker* cenderung stabil, karen tidak ada perubahan akurasi dari penggunaan 1 *node worker* hingga 4 *node worker*.

Berdasarkan hasil pengujian dengan menggunakan variasi jumlah *node worker*, semakin banyak *node* yang digunakan maka semakin sedikit juga waktu yang dibutuhkan untuk melakukan analisis sentimen ulasan pelanggan Amazon. Pada pengujian ini jumlah *node* memiliki korelasi dengan semakin cepatnya waktu proses analisis sentimen. Dalam hal ini semakin banyak *node worker* yang terlibat dalam pengolahan data, semakin banyak sumber daya komputasi yang tersedia untuk memproses tugas secara paralel.

Pada pengujian ini penggunaan 4 *node worker* menjadi jumlah *node worker* yang paling optimal dalam implementasi Naive Bayes pada Spark untuk melakukan analisis sentimen ulasan pelanggan Amazon. Dalam penggunaan 4 *node worker*, sistem dapat mengoptimalkan pemrosesan data secara distribusi, sehingga menghasilkan hasil analisis sentimen yang cukup akurat.

Dalam konteks kecepatan pemrosesan dapat dikatakan bahwa pada pengujian ini, *running time* yang dihasilkan mengalami penurunan yang cukup signifikan pada variasi jumlah *node worker* yang digunakan dalam menganalisis dan mengklasifikasikan sentimen pelanggan Amazon. Dan dalam konteks akurasi, pada pengujian model mampu mengklasifikasikan sentimen pelanggan dengan akurasi yang cukup baik yaitu 82.31% untuk dataset berukuran 500 MB, 82.42% untuk dataset berukuran 1000 MB dan 82.23% untuk dataset berukuran 1500 MB. Dengan demikian, pengujian untuk mengimplementasikan Naive Bayes pada Spark menggunakan variasi jumlah *node worker* dapat dikatakan berhasil, karena mencapai tingkat akurasi yang cukup baik dengan waktu pemrosesan yang efisien. Namun dibutuhkan penelitian lebih lanjut lagi, seperti menggunakan lebih dari 4 *node worker* agar dapat menjadi langkah yang baik untuk memperluas pemahaman tentang kinerja sistem dalam skenario yang lebih kompleks. Dan dapat mengevaluasi dampak penambahan *node worker* terhadap running time, akurasi, dan faktor-faktor lain yang relevan.

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan hasil pengujian yang telah dilakukan dapat disimpulkan yang dapat diambil adalah sebagai berikut:

1. Pada penelitian ini, dilakukan analisis sentimen ulasan pelanggan Amazon menggunakan metode Naive Bayes yang diimplementasikan dengan menggunakan *library* MLlib pada platform Spark. Hasil pengujian menunjukkan bahwa metode Naive Bayes dengan menggunakan MLlib mampu memberikan akurasi yang cukup baik sebesar 82.31% untuk dataset berukuran 500 MB, 82.42% untuk dataset berukuran 1000 MB dan 82.23% untuk dataset berukuran 1500 MB.
2. Pengujian menunjukkan bahwa jumlah *node worker* memiliki dampak signifikan terhadap waktu proses analisis sentimen. Pemilihan jumlah *node worker* yang optimal menjadi penting dalam meningkatkan kinerja sistem secara keseluruhan. Pada pengujian ini, penggunaan 4 *node worker* menghasilkan waktu eksekusi yang paling cepat. Namun, perlu dilakukan evaluasi lebih lanjut untuk mempertimbangkan faktor-faktor lain seperti sumber daya yang tersedia dan kompleksitas tugas yang dijalankan

B. Saran

Berdasarkan pengujian yang telah dilakukan terdapat beberapa saran yang dapat membantu pengembangan sistem:

1. Dalam konteks penyimpanan dataset, disarankan untuk menggunakan HDFS sebagai solusi penyimpanan. Penggunaan HDFS akan memberikan keuntungan

dalam efisiensi dan efektivitas dalam penyebaran data di dalam *cluster*.

2. Dapat menggunakan lebih dari 4 *node worker* untuk mengetahui kinerja Spark yang lebih optimal. Dengan menggunakan lebih banyak *node worker*, dapat dilakukan eksperimen yang lebih luas dan mendalam untuk mengevaluasi kinerja dan skalabilitas Spark dalam berbagai scknario pengujian.

DAFTAR PUSTAKA

[1] C. Wibawa, S. Wirawan, M. Mustikasari, and D. T. Anggraeni, "KOMPARASI KECEPATAN HADOOP MAPREDUCE DAN APACHE SPARK DALAM MENGOLAH DATA TEKS," *Jurnal Ilmiah MATRIK*, vol. 24, no. 1, 2022.

[2] I. R. Prabaswara and R. Saputra, "Analisis Data Sosial Media Twitter Menggunakan Hadoop dan Spark," *IT JOURNAL RESEARCH AND DEVELOPMENT*, vol. 4, no. 2, Mar. 2020, doi: 10.25299/itjrd.2020.vol4(2).4099.

[3] Y. K. Gupta and S. Kumari, "A study of *big data* analytics using apache spark with python and scala," in *Proceedings of the 3rd International Conference on Intelligent Sustainable Systems, ICISS 2020*, Institute of Electrical and Electronics Engineers Inc., Dec. 2020, pp. 471–478. doi: 10.1109/ICISS49785.2020.9315863.

[4] T. Tekdogan and A. Cakmak, "Benchmarking Apache Spark and Hadoop MapReduce on *Big data* Classification," 2021. doi: https://.org/10.1145/3481/646.3481.1649.

[5] Y. Samadi, M. Zbakh, and C. Tadonki, "Performance comparison between hadoop and spark frameworks using HiBench benchmarks," *Concurr Comput*, vol. 30, no. 12, Jun. 2018, doi: 10.1002/cpe.4367.

[6] E. Amaia, A. N. Izzah, A. Akram, and N. Risal, "Techno Xplore Jurnal Ilmu Komputer dan Teknologi Informasi Klasifikasi Penyalahgunaan Pesan singkat Menggunakan Algoritma Naïve Bayes," 2023. doi: http://tsel.me/tsel.

[7] X. Meng *et al.*, "MLlib: Machine Learning in Apache Spark," 2016.

[8] A. Rashid and C. Huang, "Sentiment Analysis on Consumer Reviews of Amazon Products," *International Journal of Computer Theory and Engineering*, vol. 13, no. 2, pp. 35–41, 2021, doi: 10.7763/IJCTE.2021.V13.1287.

[9] H. Nguyen *et al.*, "Comparative Study of Sentiment Analysis with Product Reviews Using Machine Learning and Lexicon-Based Approaches." [Online]. Available: https://scholar.smu.edu/datasciencereviewAvailablea t:https://scholar.smu.edu/datasciencereview/vol1/iss 4/7http://digitalrepository.smu.edu.

[10] L. D. Utami, S. Tinggi, M. Informatika, D. Komputer, N. Mandiri, and R. S. Wahono, "Integrasi Metode Information Gain Untuk Seleksi Fitur dan Adaboost Untuk Mengurangi Bias Pada Analisis Sentimen Review Restoran Menggunakan Algoritma Naïve Bayes," *Journal of Intelligent Systems*, vol. 1, no. 2, 2015, [Online]. Available: http://journal.ilmukomputer.org

[11] R. A. Fauzi, I. Cholissodin, and B. Rahayudi, "Pemanfaatan Spark untuk Analisis Sentimen Mengenai Netralitas Berita dalam Membahas Pemilu Presiden 2019 Menggunakan Metode Naïve Bayes Classifier," 2021. doi: http://j-ptiik.ub.ac.id.

[12] B. Liu, E. Blasch, Y. Chen, D. Shen, and G. Chen, "Scalable Sentiment Classification for *Big data* Analysis Using Na¨ıveNa¨ıve Bayes Classifier." [Online]. Available: http://mahout.apache.org

[13] XIAOFANG WAng, LAN LUO, ZOU QIANYIN, and etc., "CONSTRUCTING NAIVE BAYESIAN CLASSIFICATION MODEL BY SPARK FOR *BIG DATA*," *17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)* , 2020.

[14] C. Darujati and A. Bimo Gumelar, "PEMANFAATAN TEKNIK SUPERVISED UNTUK KLASIFIKASI TEKS BAHASA INDONESIA," Surabaya, Feb. 2012.

[15] D. Jeremia, H. Novianus, and A. Gunawan, "Platform *Big data* Analytic Berbasis Apache Spark Bagi Pemula Dalam Menyusun Data Analysis Workflow," Surabaya, 2022.

[16] E. Riyandani, "'*Big data* vs Big Information vs Big Knowledge' Oleh: Imam Cholissodin," 2016. doi: http://bit.ly/2x8ta9S.

[17] B. Maryanto, "*BIG DATA* DAN PEMANFAATANNYA DALAM BERBAGAI SEKTOR," Bandung, 2017.

[18] C. Kurniawan, "A Survey on *Big data* Analytics Model," *ITEJ* , vol. 4, no. 1, Jul. 2019.

[19] A. Nurzahputra and A. Muslim, *Analisis Sentimen pada Opini Mahasiswa Menggunakan Natural Language Processing*. Semarang, 2016.

[20] B. Mas Pintoko and K. Muslim, "Analisis Sentimen Jasa Transportasi Online pada Twitter Menggunakan Metode Naïve Bayes Classifier," Bandung, Dec. 2018.

[21] B. Alfironi Muktamar, A. Setiawan, and B. Adji, "Pembobotan Korelasi pada Naive Bayes Classifier," pp. 6–8, Feb. 2015.

[22] R. Fajar, S. Program, P. Rekayasa, N. Lunak, and R. Bengkalis, "Implementasi Algoritma Naive Bayes Terhadap Analisis Sentimen Opini Film Pada Twitter," *Jurnal Invotek Polbeng*, vol. 3, no. 1, Jun. 2018.