**IMPLEMENTASI ALGORITMA *K-NEAREST NEIGHBOR* DALAM SISTEM *CASE BASED REASONING* UNTUK**

**PENCARI JAWABAN DARI SOAL-SOAL**

**ALGORITMA**

**TUGAS AKHIR**

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik

pada jurusan Teknik Informatika Fakultas Sains dan Teknologi

Universitas Islam Negeri Sunan Gunung Djati Bandung

**Oleh**

**Nurida Ahsanti**

**1127050118**

****

**BANDUNG**

**2016M/1437H**

# BAB I

# PENDAHULUAN

## Latar Belakang

Di era digital ini cukup jelas membuktikan betapa pesatnya perkembangan teknologi dan informasi. Hal ini akan terus berlangsung karena semakin banyaknya minat dalam persaingan mengembangkan atau membuat berbagai teknologi baru. Pada nyatanya, bukan hanya orang yang khusus terlibat di bidang IT saja yang berpotensi dalam persaingan tersebut tetapi orang-orang di luar IT pun ikut tertarik. Oleh karena itu, pembelajaran tentang proses pembuatan perangkat lunak melalui contoh program/*coding* yang sudah ada sangat diperlukan.

Agar dapat membuat sebuah program, maka terlebih dahulu harus mengetahui bagaimana memecahkan suatu permasalahan yang ada. Karena belajar memprogram adalah belajar tentang metodologi pemecahan masalah, kemudian menuangkan algoritma pemecahan masalah dalam suatu notasi tertentu. [1] Namun pada nyatanya untuk memahami benar-benar algoritma yang nantinya akan dijadikan dasar dari pemrograman tidak dapat disepelekan walaupun hal tersebut tidak juga dikatakan sulit. Hal tersebut dapat dibuktikan dari hasil peninjauan nilai akhir mahasiswa angkatan 2012-2014 pada mata perkuliahan Algoritma dan Struktur Data di UIN Sunan Gunung Djati Bandung bahwa 28.52% diantaranya masih mempunyai hasil nilai menengah kebawah. Oleh sebab itu, masih tetap diperlukan upaya dalam peningkatan pembelajaran agar mencapai hasil yang lebih maksimal.

Pada proses pembelajaran, cara yang mudah dan biasa dilakukan ketika dihadapkan dengan soal-soal algoritma yaitu membandingkannya dengan soal-soal dan jawaban yang telah ada kemudian diperkirakan mirip atau tidaknya. Hal tersebut dilakukan dengan tanpa mengetahui apakah soal dan jawaban tersebut benar-benar mirip atau tidak, sehingga tidak jarang pula hasil jawaban yang diperoleh kurang memuaskan.

Dalam membuat notasi dari pemecahan permasalahan algoritma, diperlukan suatu identitas yang dapat mewakili pemecahan masalah untuk jawaban yang diperoleh dari soal algoritma. Dan yang perlu diketahui bahwa sebuah algoritma dapat dibangun dari tiga struktur dasar yaitu Runtunan (*Sequence*), Pemilihan (*Selection*), dan Pengulangan (*Repetition*). Sedangkan pada sebuah *coding* disusun oleh arsitektur *coding* yang terdiri dari tipe data, konstanta, variabel dan algoritma dalam bentuk notasi itu sendiri. Dalam hal ini, pembentukan identitas tersebut dapat diukur dari arsitektur *coding* dengan cara membandingkan identitas baru dengan identitas-identitas sebelumnya sehingga dapat diklasifikasikan berdasarkan struktur dasar tersebut agar mendapatkan hasil yang paling mirip menurut voting terbanyak.

Dalam hal ini sejalan dengan metode *Case Based Reasoning (CBR)*, yaitu cara pemecahan masalah dengan memanfaatkan pengalaman sebelumnya dalam domain pengetahuan tertentu. Maher et al (1995) mengungkapkan bahwa "CBR adalah sebuah pendekatan untuk masalah pemecahan yang menggunakan database atau kasus masalah sebelumnya itu diselesaikan ketika memecahkan masalah baru dimana database adalah kumpulan data yang disimpan dalam komputer". [2] Pada prosesnya *CBR* melalui 4 tahap, yaitu: *Retrieve* (mengambil masalah/kasus yang paling serupa dan berakhir apabila menemukan kecocokan dengan masalah sebelumnya yang memiliki tingkat kompatibilitas tertinggi), *Reuse* (menggunakan kembali pengetahuan dan informasi berdasarkan berat kasus lama dengan kesamaan paling relevan dengan kasus baru), *Revise* (meninjau solusi yang diusulkan dan mengujikan pada kasus nyata (simulasi) dan jika perlu meluruskan solusi tersebut untuk mencocokkan kasus baru) dan *Retain* (mempertahankan/menyimpan solusi baru sebagai bagian dari masalah/kasus baru). [3]

Sedangkan pada perhitungan tingkat kemiripan (jarak) antar identitas jawaban didasarkan pada penggunaan beberapa atribut yang terdefinisi sebelumnya, yaitu dari penjumlahan antara pembobotan dengan tingkat kemiripan (jarak) keseluruhan atribut. Pada proses ini dapat digunakan algoritma klasifikasi seperti *K-Nearest Neighbor,* *Naive bayes*, *LVQ* dan *K-Means.*

Pada hasil penelitian yang dilakukan sebelumnya bahwa algoritma *K-Nearest Neighbor (KNN)* merupakan suatu metode yang menggunakan algoritma *supervised* dimana hasil dari *query instance* yang baru diklasifikasikan berdasarkan mayoritas dari kategori yang ada sebelumnya pada *KNN*. Tujuan dari algoritma ini adalah mengklasifikasikan obyek baru bedasarkan atribut dan *training sample*.

Lain halnya dengan algoritma klasifikasi *Naive bayes* yangtermasuk *multinomial* dalam mengambil jumlah kata yang muncul pada sebuah dokumen. Pada model ini, sebuah dokumen terdiri atas beberapa kejadian kata dan diasumsikan panjang dokumen tidak bergantung pada kelasnya. Sehingga memungkinkan pada tiap kejadian kata dalam sebuah dokumen dapat bebas dan tidak terpengaruh dengan konteks kata atau posisi kata dalam dokumen.

Selain itu, algorima klasifikasi *Learning Vector Quantization (LVQ)* merupakan suatu metode untuk melakukan pembelajaran pada lapisan kompetitif terawasi yang nantinya akan secara otomatis belajar untuk mengklasifikasikan vektor-vektor input. Jika 2 vektor input mendekati sama, maka lapisan kompetitif akan meletakkan kedua vektor tersebut ke dalam kelas yang sama.

Sedangkan pada algorima klasifikasi *K-Means*, setiap data harus masuk *cluster* (kelompok) tertentu. Prinsip utama dari teknik ini adalah menyusun *k*buah prototipe atau pusat massa (*centroid*) atau pun rata-rata (*mean*) dari sekumpulan data berdimensi *n*. Teknik ini mensyaratkan nilai *k* sudah diketahui sebelumnya. [4]

Dari keempat algoritma klasifikasi yang telah dijabarkan sebelumnya dapat diambil kesimpulan bahwa algoritma *Naive bayes* memang lebih unggul dalam mengambil jumlah kata yang muncul tetapi cenderung lebih bebas dan tidak bergantung pada kelasnya. Sedangkan *Learning Vector Quantization (LVQ)* mempunyai prinsip pengklasifikasian berdasarkan vektor-vektor input sehingga kurang efektif jika pembandingnya berupa data yang banyak. Dan *K-Means* berprinsip setiap data harus masuk *cluster* (kelompok) dahulu jika tidak maka algoritma ini tidak bisa dilakukan. Lain halnya dengan *K-Nearest Neighbor* *(KNN)* yang hasil dari pengklasifikasian obyek baru didasarkan pada atribut dan *training sample* atau singkatnya mengklasifikasikan dengan voting terbanyak yang diperoleh dari perbandingan data yang telah ada sebelumnya sehingga menjadikan algoritma ini dianggap lebih cocok dengan konsep penelitian yang akan dilakukan.

Dalam mengklasifikasikan pencarian kemiripan soal dan jawaban algoritma dengan yang paling mirip merupakan solusi untuk memecahkan permasalahan dari soal algoritma. Untuk itu, maka disusunlah penelitian “**Implementasi Algoritma *K-Nearest Neighbor* Dalam *Sistem Case Based Reasoning* Untuk Pencari Jawaban dari Soal-soal Algoritma**”.

## Rumusan Masalah

Rumusan masalah yang akan dijadikan objek pada penelitian adalah sebagai berikut:

1. Bagaimana membangun sistem *Case Based Reasoning* yang dapat membentuk identitas jawaban otomatis dari soal algoritma dan mencari kemiripan jawaban tersebut menggunakan algoritma *K-Nearest Neighbor*?
2. Bagaimana penerapan algoritma *K-Nearest Neighbor* dan metode *Case Based Reasoning* dalam tingkat akurasi hasil pembentukan identitas jawaban otomatis dan pencari kemiripan jawaban?

## Tujuan

Adapun tujuan yang ingin dicapai dari penelitian ini yakni sebagai berikut:

1. Membangun sistem *Case Based Reasoning* yang dapat membentuk identitas jawaban otomatis dari soal algoritma dan mencari kemiripan tersebut menggunakan algoritma *K-Nearest Neighbor.*
2. Mengetahui penerapan algoritma *K-Nearest Neighbor* dan metode *Case Based Reasoning* dalam tingkat akurasi hasil pembentukan identitas jawaban otomatis dan pencari kemiripan jawaban.

## Batasan Masalah

Adapun batasan permasalahan dari penelitian ini yakni sebagai berikut:

1. Implementasi dari sistem ini berbasis dekstop *offline*.
2. Pembentukan identitas jawaban otomatis dilakukan agar hasilnya dapat dibandingkan kemiripannya dengan identitas jawaban yang sebelumnya.
3. Kemiripan jawaban hanya sebagai gambaran untuk mengerjakan soal-soal algoritma yang jawabannya harus menggunakan *coding.*
4. Hasil tingkat akurasi di dapat dari metode *Case Based Reasoning* dan algoritma *K-Nearest Neighbor*.

## Metodologi Penelitian

### Teknik Pengumpulan Data

Metodologi Penelitian diperlukan sebagai alat bantu untuk memudahkan pekerjaan didalam melakukan perancangan aplikasi. Metodologi penelitian yang digunakan yaitu sebagai berikut:

1. Studi Kepustakaan

Studi pustaka merupakan langkah awal dalam metode pengumpulan data. Studi pustaka merupakan metode pengumpulan data yang diarahkan kepada pencarian data dan informasi melalui jurnal penelitian, internet, buku dan *e-book* yang dapat mendukung dalam proses penulisan.”Hasil penelitian juga akan semakin kredibel apabila didukung foto-foto atau karya tulis akademik dan seni yang telah ada.” [5] Maka dapat dikatakan bahwa studi pustaka dapat memengaruhi kredibilitas hasil penelitian yang dilakukan.

1. Wawancara

Wawancara atau *interview* merupakan teknik pengumpulan data dengan cara bertatap muka secara langsung dengan informan dan dosen yang bersangkutan. Wawancara dilakukan jika data yang diperoleh kurang mendalam. Wawancara digunakan sebagai teknik pengumpulandata apabila peniliti ingin mengetahui hal-hal dari informan yang lebih mendalam. [5]

### Model Proses Pengembangan Perangkat Lunak

Adapun untuk pembuatan aplikasi ini, menggunakan metode pengembangan perangkat lunak *Prototype,* karena metode ini lebih memudahkan dalam proses membangun sebuah perancangan aplikasi. Alur dari metode *Prototype* terdiri dari langkah-langkah sebagai berikut:

1. *Communication,* yaitu *Developer* dan klien bertemu dan menentukan tujuan umum, kebutuhan yang diinginkan dan gambaran bagian-bagian yang akan dibutuhkan berikutnya.
2. *Quick Plan* atau Perancangan yang dilakukan dengan cepat dan mewakili semua aspek *software* yang diketahui, dan rancangan ini menjadi dasar pembuatan *Prototype*.
3. *Modelling Quick Design*, langkah ini berfokus pada representasi aspek *software* yang bisa dilihat *customer/User* dan cenderung ke pembuatan *prototype*.
4. *Construction of Prototype,* yaitu membangun kerangka atau rancangan *prototype* dari *software* yang akan dibangun.
5. *Deployment Delivery & Feedback* merupakan *Prototype* yang telah dibuat oleh *developer* akan disebarkan kepada *User*/klien untuk dievaluasi, kemudian klien akan memberikan *feedback* yang akan digunakan untuk merevisi kebutuhan *software* yang akan dibangun. Pengulangan proses ini terus berlangsung sampai semua kebutuhan terpenuhi. [6]

## Sistematika Penulisan

Sistematika penulisan laporan tugas akhir disusun dalam beberapa bab yang masing-masing bab menguraikan beberapa pokok pembahasan. Adapun sistematika penulisan laporan ini adalah sebagai berikut:

### BAB 1 PENDAHULUAN

Bab ini terdiri dari enam subbab yaitu latar belakang menggambarkan hal-hal yang perlu dijadikan penelitian dari realita yang ada, rumusan masalah, tujuan, batasan masalah, metodologi penelitian, dan sistematika penulisan. Subbab Latar belakang masalah menggambarkan hal-hal yang perlu dijadikan penelitian dari realita yang ada. Subbab Rumusan masalah dituliskan kedalam bentuk poin yang menjadi sasaran utama pada objek yang akan diteliti. Subbab Tujuan menggambarkan hal-hal yang ingin dicapai. Subbab Batasan masalah berisi batasan yang ditentukan dalam perancangan sistem berupa hal-hal terkait dengan sistem. Subbab Metodologi penelitian terbagi kedalam dua tahapan yaitu teknik pengumpulan data dan menggambarkan model proses pengembangan perangkat lunak yang dibuat. Dan bagian terakhir dari bab ini yakni dengan subbab sistematika penulisan yang menguraikan urutan penyajian yang digunakan dalam penyusunan skripsi.

### BAB II LANDASAN TEORI

Bab ini menjelaskan teori-teori yang digunakan dalam menganalisis permasalahan yang ada dan teori-teori yang akan digunakan untuk membangun perangkat lunak ini.

### BAB III ANALISIS DAN PERANCANGAN

Pada bagian ini akan menguraikan hasil analisis dan perancangan perangkat lunak yang akan dibagun.

### BAB IV IMPLEMENTASI

Pada bab ini menjelaskan tentang spesifikasi perangkat lunak, kebutuhan perangkat lunak, implementasi perangkat lunak, dan pengujian yang dilakukan terhadap perangkat lunak yang dibangun.

### BAB V PENUTUP

Berisi tentang pernyataan singkat berupa kesimpulan dari pembahasan perangkat lunak yang dibuat secara keseluruhan dan saran untuk mengembangkan perangkat lunak yang lebih baik.

# BAB II

# LANDASAN TEORI

## Algoritma

Algoritma muncul dari Algoritmi, bentuk Latin dari [al-Khwarizmi](https://id.wikipedia.org/w/index.php?title=Muhammad_ibnu_Musa_al-Khwarizmi&action=edit&redlink=1), [matematikawan](https://id.wikipedia.org/w/index.php?title=Matematikawan_islam&action=edit&redlink=1), [ahli astronomi](https://id.wikipedia.org/w/index.php?title=Astronomi_islam&action=edit&redlink=1), dan [ahli geografi](https://id.wikipedia.org/w/index.php?title=Ahli_geografi_islam&action=edit&redlink=1) dari [Persia](https://id.wikipedia.org/wiki/Orang_Persia). [7] Untuk masalah dengan instansiasi yang kecil, solusi dapat ditemukan dengan mudah dan cepat. Lain halnya dengan instansiasi masalah yang berukuran besar, jelas tidak mudah mengurutkan data sebanyak itu. Oleh karena itu, perlu untuk menuliskan prosedur yang berisi langkah-langkah pengurutan sehingga prosedur tersebut dapat dijalankan oleh sebuah pemroses (komputer, manusia, robot, dan sebagainya) untuk menghasilkan solusi setiap instansiasi masalah pengurutan. Prosedur yang berisi langkah-langkah penyelesaian masalah tersebut disebut “algoritma”. Algoritma adalah urutan langkah-langkah untuk memecahkan suatu masalah. [1] Dalam [matematika](https://id.wikipedia.org/wiki/Matematika) dan [ilmu komputer](https://id.wikipedia.org/wiki/Ilmu_komputer), algoritma adalah prosedur langkah-demi-langkah untuk penghitungan. Algoritma digunakan untuk [penghitungan](https://id.wikipedia.org/wiki/Penghitungan), [pemrosesan data](https://id.wikipedia.org/wiki/Pemrosesan_data), dan [penalaran otomatis](https://id.wikipedia.org/w/index.php?title=Penalaran_otomatis&action=edit&redlink=1).

Algoritma adalah [metode efektif](https://id.wikipedia.org/w/index.php?title=Metode_efektif&action=edit&redlink=1) diekspresikan sebagai rangkaian [terbatas](https://id.wikipedia.org/w/index.php?title=Terbatas&action=edit&redlink=1) dari instruksi-instruksi yang telah didefinisikan dengan baik untuk menghitung sebuah [fungsi](https://id.wikipedia.org/wiki/Fungsi_%28matematika%29). Dimulai dari sebuah kondisi awal dan input awal, instruksi-instruksi tersebut menjelaskan sebuah [komputasi](https://id.wikipedia.org/wiki/Komputasi) yang bila [dieksekusi](https://id.wikipedia.org/w/index.php?title=Eksekusi_%28komputasi%29&action=edit&redlink=1) prosesnya melalui sejumlah urutan kondisi terbatas yang terdefinisi dengan baik, sehingga pada akhirnya menghasilkan "keluaran" dan berhenti di kondisi akhir. [[6]](https://id.wikipedia.org/wiki/Algoritma#cite_note-6) Transisi dari satu kondisi ke kondisi selanjutnya tidak harus [deterministik](https://id.wikipedia.org/w/index.php?title=Deterministik&action=edit&redlink=1). Beberapa algoritma dikenal dengan [algoritma pengacakan](https://id.wikipedia.org/w/index.php?title=Algoritma_pengacakan&action=edit&redlink=1) karena menggunakan masukan acak.

Walaupun [*algorism*](https://id.wikipedia.org/wiki/Algorism)-nya [al-Khawarizmi](https://id.wikipedia.org/wiki/Al-Khawarizmi) dirujuk sebagai aturan-aturan melakukan aritmetika menggunakan [bilangan Hindu-Arab](https://id.wikipedia.org/w/index.php?title=Bilangan_Hindu-Arab&action=edit&redlink=1) dan solusi sistematis dan [persamaan kuadrat](https://id.wikipedia.org/wiki/Persamaan_kuadrat), sebagian formalisasi yang nantinya menjadi *algoritma* modern dimulai dengan usaha untuk memecahkan [permasalahan keputusan](https://id.wikipedia.org/w/index.php?title=Permasalahan_keputusan&action=edit&redlink=1) (*Entscheidungsproblem*) yang diajukan oleh [David Hilbert](https://id.wikipedia.org/wiki/David_Hilbert) pada tahun 1928. Formalisasi selanjutnya dilihat sebagai usaha untuk menentukan "[penghitungan efektif](https://id.wikipedia.org/w/index.php?title=Penghitungan_efektif&action=edit&redlink=1)" atau "metode efektif". Formalisasi tersebut mengikutkan [Godel](https://id.wikipedia.org/w/index.php?title=Kurt_Godel&action=edit&redlink=1), [Herbrand](https://id.wikipedia.org/w/index.php?title=Jacques_Herbrand&action=edit&redlink=1), [Kleene](https://id.wikipedia.org/w/index.php?title=Stephen_Cole_Kleene&action=edit&redlink=1) dengan [fungsi rekursif](https://id.wikipedia.org/w/index.php?title=Rekursi_%28ilmu_komputer%29&action=edit&redlink=1) dari [Kurt Godel](https://id.wikipedia.org/w/index.php?title=Kurt_Godel&action=edit&redlink=1), [Jacques Herbrand](https://id.wikipedia.org/w/index.php?title=Jacques_Herbrand&action=edit&redlink=1), [Stephen Cole Kleene](https://id.wikipedia.org/w/index.php?title=Stephen_Cole_Kleene&action=edit&redlink=1) pada tahun 1930, 1934, dan 1935, kemudian [kalkulus lambda](https://id.wikipedia.org/w/index.php?title=Kalkulus_lambda&action=edit&redlink=1) oleh [Alonzo Church](https://id.wikipedia.org/w/index.php?title=Alonzo_Church&action=edit&redlink=1) pada tahun 1936, "[Formulasi 1](https://id.wikipedia.org/w/index.php?title=Formulasi_1&action=edit&redlink=1)" oleh [Emil Post](https://id.wikipedia.org/w/index.php?title=Emil_Post&action=edit&redlink=1) pada tahun 1936, dan [Mesin Turing](https://id.wikipedia.org/wiki/Mesin_Turing) oleh [Alan Turing](https://id.wikipedia.org/wiki/Alan_Turing) pada tahun 1936 sampai dengan tahun 1937 dan pada tahun 1939. Dari definisi formal dari algoritma di atas, berkaitan dengan konsep intuitif, masih tetap ada masalah yang menantang. [7]

Dalam pemrograman yang sederhana, algoritma merupakan langkah pertama yang harus ditulis sebelum menuliskan program. Masalah yang dapat diselesaikan dengan pemrograman komputer adalah masalah-masalah yang berhubungan dengan perhitungan matematik. Hal yang penting dikuasai dalam pemrograman adalah logika berpikir bagaimana cara memecahkan masalah pemrograman yang akan dibuat. Kadang-kadang ada masalah matematika yang sangat gampang jika diselesaikan secara tertulis, tetapi cukup sulit jika diterjemahkan ke dalam pemrograman. Jika menemukan hal seperti itu, maka algoritma dan logika pemrograman sangat penting untuk memecahkan masalah. [8]

Algoritma adalah jantung ilmu komputer atau informatika. Banyak cabang ilmu komputer yang diacu dalam terminologi algoritma. Dalam kehidupan sehari-haripun banyak terdapat proses yang digambarkan dalam suatu algoritma. Sebuah algoritma merupakan deskripsi langkah-langkah pelaksanaan suatu proses. Setiap langkah di dalam algoritma dinyatakan dalam sebuah pernyataan (*statement*) atau istilah lainnya instruksi. Sebuah pernyataan berisi aksi (*action*) yang dilakukan. Bila sebuah pernyataan dieksekusi oleh pemroses, maka aksi yang bersesuaian dengan pernyataan itu dikerjakan. [1]

### Struktur Dasar Algoritma

Di dalam algoritma terdapat beberapa jenis pernyataan, seperti pernyataan ekspresi, pernyataan pemilihan, pernyataan pengulangan, pernyataan prosedur, pernyataan gabungan, dan sebagainya. Algoritma berisi langkah-langkah penyelesaian masalah. Langkah-langkah tersebut dapat berupa runtunan aksi, pemilihan aksi, dan pengulangan aksi. Ketiga jenis langkah tersebut membentuk konstruksi suatu algoritma. Jadi, sebuah algoritma dapat dibangun dari tiga buah konstruksi atau struktur dasar, yaitu runtunan (*sequence*), pemilihan (*selection*), dan pengulangan (*repetition*).

Sebuah runtunan terdiri dari satu atau lebih pernyataan, setiap pernyataan ditulis dalam satu baris atau dipisahkan dengan tanda titik koma. Tiap pernyataan dikerjakan secara berurutan (sekuensial) sesuai dengan urutan di dalam teks algoritma, yaitu sebuah instruksi dilaksanakan setelah instruksi sebelumnya selesai dilaksanakan. Urutan instruksi menentukan keadaan akhir algoritma. Bila urutannya diubah, maka hasil akhirnya mungkin juga akan berubah. Runtunan satu atau lebih pernyataan dinamakan “pernyataan gabungan” (compound statements).

Selanjutnya tentang struktur pemilihan *if-then,* struktur ini hanya memberikan satu pilihan aksi bila kondisi (persyaratan) dipenuhi (bernilai benar), dan tidak memberi pilihan aksi lain bila kondisi bernilai salah. Kelebihan struktur pemilihan terletak pada kemampuannya yang memungkinkan memproses mengikuti jalur aksi yang berbeda berdasarkan kondisi yang ada. Tanpa struktur pemilihan, maka memungkinkan tidak dapat menuliskan algoritma untuk perrmasalahan praktis yang demikian kompleks.

Sedangkan untuk pengulangan, di dalam algoritma terdapat banyak notasi pengulangan yang dapat digunakan, antara lain *repeat N times, for, repeat-until*, dan *while.* Notasi pertama, *repeat N times* yang artinya ulangi sebanyak N kali. Notasi kedua, struktur pengulangan yang mirip dengan *repeat N times* adalah *for* yang artinya aksi dilakukan sebanyak hitungan cacah pengulangan, yaitu dari 1 sampai *N* (yaitu sebanyak *N*  kali). Pencacah pengulangan dapat di atur tidak hanya mulai dari 1, tetapi juga dari sembarang nilai yang lain. Struktur pengulangan yang ketiga adalah *repeat-until* (*repeat* artinya “ulangi” dan *until* artinya “sampai” atau “hingga”), pengulangan dilakukan hingga kondisi (persyaratan) berhenti terpenuhi. Struktur pengulangan yang terakhir adalah *while* artinya “selagi” atau “selama”. Yaitu selama kondisi (persyaratan) pengulangan masih benar, maka aksi dikerjakan. Perbedaannya dengan *repeat-until,* jika ada *repeat-until* kondisi pengulangan dievaluasi di akhir, sedangkan pada *while-do* kondisi pengulangan dievaluasi di awal pengulangan. [1]

### Notasi Algoritmik

Algoritma berisi deskripsi langkah-langkah penyelesaian masalah. Langkah-langkah penyelesaian tersebut dapat dituliskan dalam notasi algoritmik sembarang, asalkan mudah dibaca dan dipahami. Tidak ada notasi yang standar untuk menuliskan algoritma sebagaimana pada notasi bahasa pemrograman. Setiap orang boleh saja mendefinisikan notasi algoritmiknya sendiri. Hal ini bukan persoalan karena notasi algoritma tidak sama dengan kode program komputer. Program komputer adalah implementasi algoritma dalam notasi bahasa pemrograman tertentu.

Notasi algoritmik yang baik adalah notasi tang mudah dibaca dan mudah pula ditranslasikan kedalam bahasa pemrograman. Notasi algoritmik berupa *pseudo-code* mempunyai korespondensi dengan notasi bahasa pemrograman sehingga proses penerjemah dari *pseudo-code* ke kode program menjadi lebih mudah. Tidak ada aturan baku membuat *pseudo-code*. Tidak seperti bahasa pemrograman yang direpotkan dengan tanda titik koma (*semicolon*), indeks, format keluaran, kata-kata khusus, dan sebagainya. Sembarang versi *pseudo-code* dapat diterima asalkan notasinya bisa dipahami.

Lain halnya dengan struktur teks algoritma yang selalu disusun oleh tiga bagian (blok), yaitu:

1. Bagian Judul (*header*)

Judul adalah bagian yang terdiri atas nama program dan penjelasan (spesifikasi) tentang program tersebut. Judul diawali dengan kata kunci program dan nama program X. Kata program ini bukan menyatakan program dalam bahasa komputer, tetapi menyatakan bahwa sedang menulis algoritma untuk program pemecahan masalah.

Nama program sebaiknya singkat namun cukup menggambarkan apa yang dilakukan oleh program. Di bawah nama program sebaiknya disertai dengan penjelasan singkat tentang apa masalah diprogam dan apa masukan dan keluarannya.

1. Bagian deklarasi (*declaration*)

Bagian ini digunakan untuk mengumumkan semua nama yang dipakai di dalam algoritma beserta propertinya (misalkan tipe). Nama terseebut dapat berupa nama kontanta, nama peubah, nama tipe, nama prosedur dan nama fungsi. Semua nama yang dipakai di dalam algoritma harus dikenali sebelummereka digunakan. Bagian deklarasi mungkin kosong jika tidak ada penggunaan nama.

1. Bagian algoritma

Inilah bagian inti dari sebuah program. Bagian ini berisi instruksi-instruksi pemecahan masalah dalamnotasi *pseeudo-code.*[1]

## Case Based Reasoning (CBR)

Penalaran berbasis kasus (CBR) adalah cara pemecahan masalah dengan memanfaatkan pengalaman sebelumnya pada domain pengetahuan tertentu. Maher et al (1995) mengungkapkan bahwa "CBR adalah sebuah pendekatan untuk masalah pemecahan yang menggunakan database atau kasus masalah sebelumnya itu diselesaikan ketika memecahkan masalah baru dimana database adalah kumpulan data yang disimpan dalam komputer ". [2]

*Case Based Reasoning* (CBR) suatu model penalaran yang penggabungkan pemecahan masalah, pemahaman dan pembelajaran serta memadukan keseluruhannya dengan pemrosesan memori. Tugas tersebut dilakukan dengan memanfaatkan kasus yang pernah dialami oleh sistem, yang mana kasus merupakan pengetahuan dalam konteks tertentu yang mewakili suatu pengalaman yang menjadi dasar pembelajaran untuk mencapai tujuan sistem. Atau dalam definisi lain CBR merupakan metode pemecahan masalah/kasus baru dengan melakukan adaptasi terhadap metode yang digunakan untuk memecahkan masalah/kasus lama. [3]

Secara umum siklus proses CBR adalah sebagai berikut:

1. Retrieve / Mengambil

Mendapatkan / mendapatkan kembali yang paling menyerupai / kasus yang relevan (mirip) dengan kasus baru. Fase pengambilan dimulai dengan menggambarkan / menguraikan beberapa masalah, dan berakhir ketika menemukan kecocokan untuk masalah sebelumnya yang memiliki tingkat kompatibilitas tertinggi. Bagian ini mengacu pada istilah dari identifikasi, pencocokan awal, pencarian, penyeleksian dan pelaksanaan.

1. Reuse / Menggunakan kembali

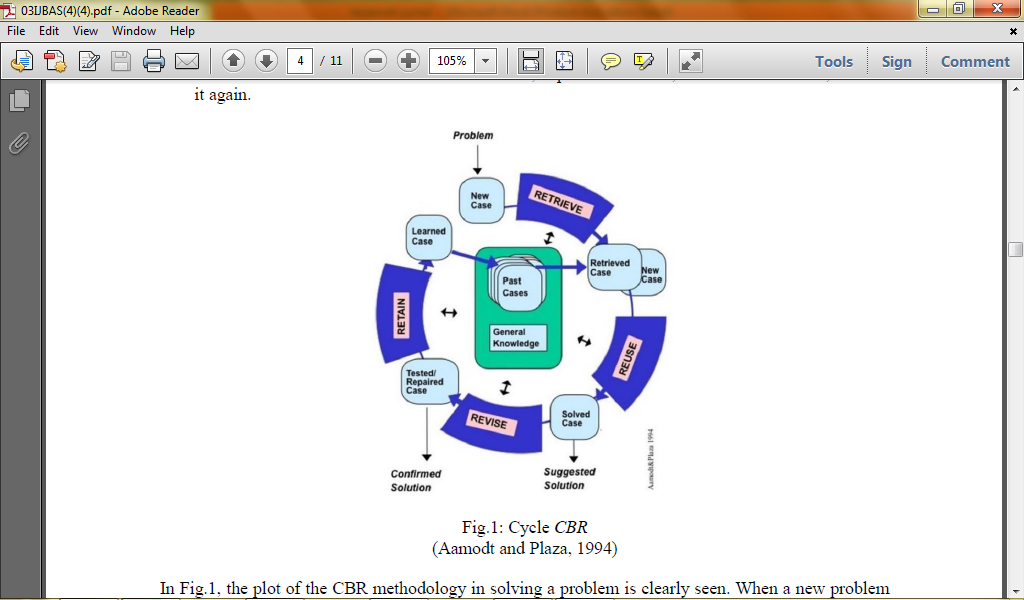
Memperagakan / menggunakan kembali pengetahuan dan informasi berdasarkan bobot kasus lama yang memiliki kesamaan paling relevan dengan kasus baru, menghasilkan pada sekumpulan solusi yang mungkin diperlukan untuk adaptasi dengan masalah baru.

3) Revise / Merevisi

Meninjau solusi yang diusulkan dan mengujinya pada kasus nyata (simulasi) dan jika perlu meluruskan solusi ini untuk mencocokkan kasus baru.

4) Retain / Mempertahankan

Mengintegrasikan / menyimpan kasus baru yang telah mendapat solusi agar dapat digunakan oleh kasus berikutnya mirip dengan kasus ini. Tetapi jika solusi baru gagal, akan menjelaskan kegagalannya kemudian memperbaiki solusi yang digunakan dan menguji lagi.



Gambar 6. Siklus CBR

Pada gambar diatas, terdapat plot metodologi CBR yang jelas dalam memecahkan suatu permasalahan. Ketika masalah baru datang, pertama-tama sistem akan melakukan proses Retrieve/ambil. Proses tersebut akan melakukan dua langkah pengolahan, yaitu pengenalan dari masalah dan mencari persamaan pokok masalah pada Database. Setelah proses Retrieve selesai, sistem akan melakukan proses Reuse. Dalam Proses reuse, sistem akan menggunakan informasi sebelumnya yang memiliki masalah yang sama untuk memecahkan masalah baru. Proses Reuse akan menyalin, memilih, dan melengkapi informasi yang akan digunakan. Kemudian pada proses Revise, informasi akan dihitung, dievaluasi dan diperbaiki untuk mengatasi kesalahan yang terjadi pada masalah baru. Dalam proses akhir, sistem akan melakukan proses Retain/Mempertahankan. Proses mempertahankan pada indeks, mengintegrasikan, dan mengekstrak solusi baru. Selain itu, solusi baru akan disetorkan ke pengetahuan dasar untuk memecahkan masalah yang akan datang. Tentu saja, masalah yang akan dipecahkan adalah masalah yang memiliki kesamaan dengan yang sebelumnya. [2]

## Data Mining

Data Mining adalah serangkaian proses untuk menggali nilai tambah dari suatu kumpulan data berupa pengetahuan yang selama ini tidak diketahui secara manual. Patut diingat bahwa kata mining sendiri berarti usaha untuk mendapatkan sedikit barang berharga dari sejumlah besar material dasar. Karena itu Data Mining sebenarnya memiliki akar yang panjang dari bidang ilmu seperti kecerdasan buatan (artificial intelligent), machine learning, statistik dan database.

Data mining adalah proses menerapkan metode ini untuk data dengan maksud untuk mengungkap pola-pola tersembunyi. Dengan arti lain Data mining adalah proses untuk penggalian pola-pola dari data. Data mining menjadi alat yang semakin penting untuk mengubah data tersebut menjadi informasi. Hal ini sering digunakan dalam berbagai praktek profil, seperti pemasaran, pengawasan, penipuan deteksi dan penemuan ilmiah. Telah digunakan selama bertahun-tahun oleh bisnis, ilmuwan dan pemerintah untuk menyaring volume data seperti catatan perjalanan penumpang penerbangan, data sensus dan supermarket scanner data untuk menghasilkan laporan riset pasar.

Alasan utama untuk menggunakan data mining adalah untuk membantu dalam analisis koleksi pengamatan perilaku. Data tersebut rentan terhadap collinearity karena diketahui keterkaitan. Fakta yang tak terelakkan data mining adalah bahwa subset/set data yang dianalisis mungkin tidak mewakili seluruh domain, dan karenanya tidak boleh berisi contoh-contoh hubungan kritis tertentu dan perilaku yang ada di bagian lain dari domain. Untuk mengatasi masalah semacam ini, analisis dapat ditambah menggunakan berbasis percobaan dan pendekatan lain, seperti Choice Modelling untuk data yang dihasilkan manusia. Dalam situasi ini, yang melekat dapat berupa korelasi dikontrol untuk, atau dihapus sama sekali, selama konstruksi desain eksperimental.

Beberapa teknik yang sering disebut-sebut dalam literatur Data Mining dalam penerapannya antara lain: clustering, classification, association rule mining, neural network, genetic algorithm dan lain-lain. Yang membedakan persepsi terhadap Data Mining adalah perkembangan teknik-teknik Data Mining untuk aplikasi pada database skala besar. Sebelum populernya Data Mining, teknik-teknik tersebut hanya dapat dipakai untuk data skala kecil saja. [9]

## [Klasifikasi, Association, dan Clustering Dalam Data Mining](http://vavajo.blogspot.com/2013/03/klasifikasi-association-dan-clustering.html)

Metode Clustering melibatkan penciptaan cluster data yang berbeda tergantung pada kedekatan atau hubungan antara data dan membentuk tema. Sebagai contoh sebuah situs web e-commerce dapat membuat cluster data tergantung pada wilayah, demografi jenis kelamin, atau kekuasaan bahkan membeli.

Metode klasifikasi mengacu pada pembentukan kelompok data dengan menerapkan algoritma dikenal ke gudang data di bawah pemeriksaan. Metode ini berguna untuk proses bisnis yang membutuhkan informasi kategoris seperti pemasaran atau penjualan. Hal ini dapat menggunakan berbagai algoritma seperti sebagai tetangga terdekat, pohon keputusan belajar dan lain-lain.

Teknik Association adalah metode data mining digunakan secara luas dan mengarah ke penemuan yang menarik hubungan antara variabel-variabel seperti yang ditemukan di gudang data di bawah pemeriksaan. Data penambang menetapkan formula yang disebut "aturan asosiasi". Dia kemudian memprediksi model masa depan dan bertindak berdasarkan model untuk mendapatkan informasi penting. Sebagai contoh mengambil kasus gelar akademik untuk spesialisasi. Jika seorang mahasiswa opt-in untuk program tertentu maka mungkin ada probabilitas tinggi bahwa ia juga dapat memilih spesialisasi yang relevan di masa depan untuk meningkatkan peluang karirnya. [10]

## K-Nearest Neighbor (KNN)

*K-Nearest Neighbor* (KNN) adalah suatu metode yang menggunakan algoritma *supervised* dimana hasil dari *query instance* yang baru diklasifikan berdasarkan mayoritas dari kategori pada KNN. Tujuan dari algoritma ini adalah mengklasifikasikan obyek baru bedasarkan atribut dan *training sample*. *Classifier* tidak menggunakan model apapun untuk dicocokkan dan hanya berdasarkan pada memori. Diberikan titik *query,* akan ditemukan sejumlah *k* obyek atau (titik *training*) yang paling dekat dengan titik *query*. Klasifikasi menggunakan *voting* terbanyak diantara klasifikasi dari k obyek. algoritma KNN menggunakan klasifikasi ketetanggaan sebagai nilai prediksi dari *query instance* yang baru.

Algoritma metode KNN sangatlah sederhana, bekerja berdasarkan jarak terpendek dari *query instance* ke *training sample* untuk menentukan KNN-nya. *Training sample* diproyeksikan ke ruang berdimensi banyak, dimana masing-masing dimensi merepresentasikan fitur dari data. Ruang ini dibagi menjadi bagian-bagian berdasarkan klasifikasi *training sample*. Sebuah titik pada ruang ini ditandai kelac *c* jika kelas *c* merupakan klasifikasi yang paling banyak ditemui pada *k* buah tetangga terdekat dari titik tersebut. Dekat atau jauhnya tetangga biasanya dihitung berdasarkan *Euclidean Distance* yang direpresentasikan sebagai berikut :

[https://kuliahinformatika.files.wordpress.com/2010/02/evans-blog-klasifikasi-gigi-molar-premolar-pada-dental-panoramic-radiograph-persamaan-euclidean-distance.jpg?w=510](https://kuliahinformatika.files.wordpress.com/2010/02/evans-blog-klasifikasi-gigi-molar-premolar-pada-dental-panoramic-radiograph-persamaan-euclidean-distance.jpg)

dimana matriks *D(a,b)* adalah jarak skalar dari kedua vektor *a* dan *b* dari matriks dengan ukuran *d* dimensi.

Pada fase *training*, algoritma ini hanya melakukan penyimpanan vektor-vektor fitur dan klasifikasi data *training sample.* Pada fase klasifikasi, fitur-fitur yang sama dihitung untuk *testing data* (yang klasifikasinya tidak diketahui).  Jarak dari vektor baru yang ini terhadap seluruh vektor *training sample* dihitung dan sejumlah *k* buah yang paling dekat diambil. Titik yang baru klasifikasinya diprediksikan termasuk pada klasifikasi terbanyak dari titik-titik tersebut.

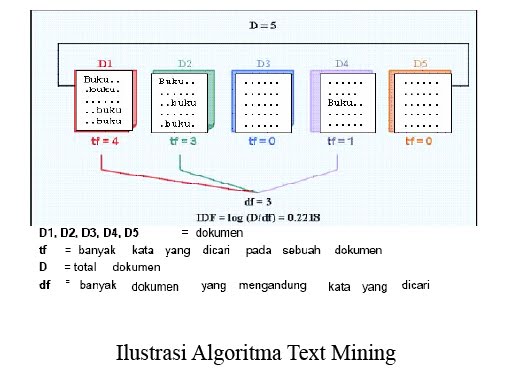
Ketepatan algoritma KNNsangat dipengaruhi oleh ada atau tidaknya fitur-fitur yang tidak relevan atau jika bobot fitur tersebut tidak setara dengan relevansinya terhadap klasifikasi. Riset terhadap algoritma ini sebagian besar membahas bagaimana memilih dan memberi bobot terhadap fitur agar performa klasifikasi menjadi lebih baik.

KNN memiliki beberapa kelebihan yaitu ketangguhan terhadap training data yang memiliki banyak noise dan efektif apabila training data-nya besar. Sedangkan, kelemahan KNN adalah KNN perlu menentukan nilai dari parameter k (jumlah dari tetangga terdekat), training berdasarkan jarak tidak jelas mengenai jenis jarak apa yang harus digunakan dan atribut mana yang harus digunakan untuk mendapatkan hasil terbaik, dan biaya komputasi cukup tinggi karena diperlukan perhitungan jarak dari tiap query instance pada keseluruhan training sample. [11]

## Text Mining

Text mining adalah salah satu bidang khusus dari *data mining*. Text mining dapat didefinisikan sebagai suatu proses menggali informasi dimana seorang user berinteraksi dengan sekumpulan dokumen menggunakan *tools* analisis yang merupakan komponen-komponen dalam *data mining* yang salah satunya adalah kategorisasi. Adapun tugas khusus dari *text mining* antara lain yaitu pengkategorisasian teks *(text categorization)* dan pengelompokan teks *(text clustering)*. [12]

*Text* *mining*  muncul karena sekitar 90% data di dunia dalam bentuk format tidak terstruktur, adanya kebutuhan bisnis, yang asalnya *document retrieval* menjadi *knowledge discovery.* Definisi *text mining* adalah proses untuk menemukan pengetahuan baru, yang belum pernah diketahui, secara otomatis oleh komputer dari sumber-sumber tertulis yang berbeda. [13]

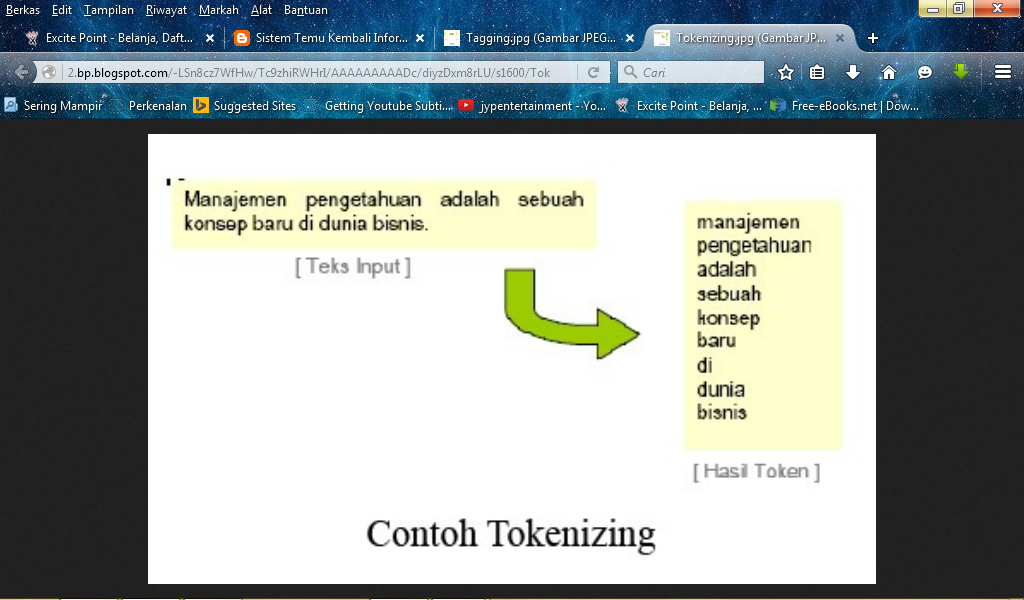


Gambar 5. Ilustrasi algoritma text mining [14]

Secara umum dalam *text mining* pada dokumen atau suatu teks dilakukan tahap sebagai berikut:

1. Tokenisasi

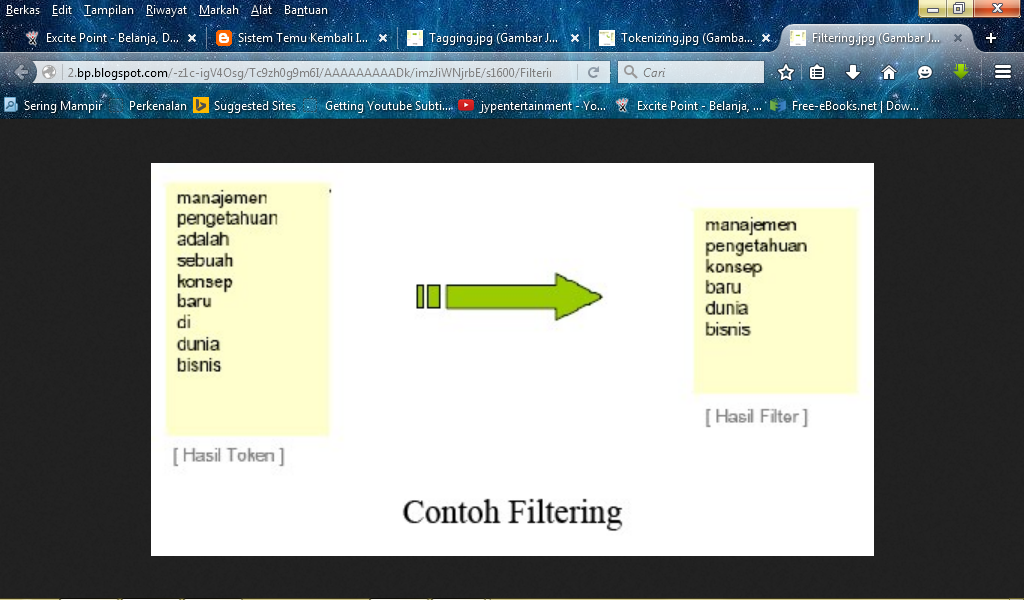
Proses *tokenizing* adalah tahap pemotongan string input berdasarkan tiap kata yang menyusunnya. Proses ini menghasilkan kata-kata yang berdiri sendiri. [15]



Gambar 1. Contoh Tokenisasi [14]

1. *Filtering*

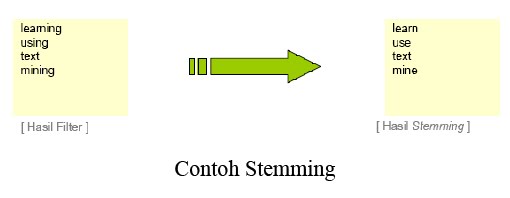
Proses *filtering* adalah tahap mengambil kata-kata penting dari hasil token. Bisa menggunakan algoritma stop list ( membuang kata-kata yang kurang penting atau word list. Proses ini akan dihasilkan kata yang penting saja dan membuang kata kata yang kurang penting. [15]



Gambar 2. Contoh *Filtering* [14]

1. *Stemming*

Proses *Stemming* adalah proses untuk mengubah bentuk kata menjadi kata dasar. Cara kerjanya adalah dengan membuang imbuhan, sisipan, dan akhiran. Tujuan proses *stemming* diantaranya adalah meningkatkan efisiensi sistem. [16]



Gambar 3. Contoh *Stemming* [14]

1. *Tagging*

Tahap tagging adalah tahap mencari bentuk awal/ root dari tiap kata lampau atau kata hasil stemming. [15]



Gambar 4. Contoh *Tagging* [14]

1. *Analyzing*

Tahap analyzing merupakan tahap penentuan seberapa jauh keterhubungan antar kata-kata antar dokumen yang ada. [15]

# DAFTAR PUSTAKA

x

|  |  |
| --- | --- |
| [1] | Munir R. Algoritma dan Pemrograman dalam Bahasa Pascal dan C, Edisi ke-3, Buku 1 Bandung: Informatika Bandung; 2011. |
| [2] | Gerhana YA, Djohar A. Case-based Reasoning Learning Model to Develop Skill in Problem Solving of Student of Vocational Education. International Journal of Basic and Applied Science. 2016 April; 04(11). |
| [3] | Luthfi ET. Penerapan Case Based Reasoning dalam Mendukung Penyelesaian Kasus. JURNA DASI. 2010;: p. 10. |
| [4] | Santoso D, Ratnawati DE, Indriati. Perbandingan Kinerja Metode Naïve Bayes, K-Nearest Neighbor, Dan Metode Gabungan K-Means Dan Lvq Dalam Pengkategorian Buku Komputer Berbahasa Indonesia Berdasarkan Judul Dan Sinopsis. Jurnal Universitas Brawijata. 2014;: p. 14. |
| [5] | Sugiyono. Metode Penelitian Kualitatif Bandung: Alfabeta; 2005. |
| [6] | Pressman RS. Rekayasa Perangkat Lunak Yogyakarta: Andi; 2002. |
| [7] | Wikipedia. Algoritma. [Online].; 2016 [cited 2016 April 3. Available from: <https://id.wikipedia.org/wiki/Algoritma>. |
| [8] | Patabang S. Pengertian Algoritma Pemrograman dan Contohnya. [Online].; 2016 [cited 2014 Januari. Available from: <http://spatabang.blogspot.co.id/2014/01/pengertian-algoritma-pemrograman-dan.html>. |
| [9] | Margaretta. Pengertian data mining. [Online].; 2016 [cited 2010 Maret 17. Available from: <http://garethdata.blogspot.co.id/2010/03/pengertian-data-mining.html>. |
| [10] | Vava BBV. Klasifikasi, Association, dan Clustering Dalam Data Mining. [Online].; 2016 [cited 2013 Maret 8. Available from: <http://vavajo.blogspot.co.id/2013/03/klasifikasi-association-dan-clustering.html>. |
| [11] | Yovianto E. Buku TA : K-Nearest Neighbor (KNN). [Online].; 2010 [cited 2016 Januari 9. Available from: <https://kuliahinformatika.wordpress.com/2010/02/13/buku-ta-k-nearest-neighbor-knn/>. |
| [12] | Diaz R. Pengertian Data Mining,Teks Mining,dan Web Mining. [Online].; 2013 [cited 2016 Januari 10. Available from: <http://yosephoriolryandiaz.blogspot.co.id/2013/03/pengertian-data-miningteks-miningdan.html>. |
| [13] | Wildyastuti H. Apakah Text Mining itu? [Online].; 2009 [cited 2016 Februari 9. Available from: <https://hildaw.wordpress.com/2009/03/12/apakah-text-mining-itu/>. |
| [14] | Vemby. Beranda: Vemby. [Online].; 2011 [cited 2016 Januari 27. Available from: <http://vemby-yoel.blogspot.co.id/2011/05/text-mining.html>. |
| [15] | Hasanah U. Label: analyzing, dokumen, filtering, kata, proses filtering, stemming, tagging. [Online].; 2012 [cited 2016 Januari 27. Available from: <http://sistemtemukembaliinformasi.blogspot.co.id/2012/07/tokenisasi.html>. |
| [16] | Rizki AS, Indriati , Muflikhah L. Text Mining Klasifikasi Soal Biologi Sekolah Menengah Atas Dengan Metode Improved KNN. Repositori Jurnal Mahasiswa PTIIK UB. 2014;: p. 8. |

x