REKOMENDASI PENGAMBILAN TEMA JUDUL SKRIPSI MENGGUNAKAN PSO-NEIGHBOR WEIGHTED K-NEAREST NEIGHBOR (NWKNN)

(STUDI KASUS: JURUSAN ILMU KEOLAHRAGAAN FAKULTAS ILMU KEOLAHRAGAAN UNIVERSITAS NEGERI MEDAN)

SKRIPSI

Disusun oleh:

Rizky Ramadhan

155150200111059



PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNIK INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2020

PENGESAHAN

REKOMENDASI PENGAMBILAN TEMA JUDUL SKRIPSI MENGGUNAKAN PSO-NEIGHBOR WEIGHTED K-NEAREST NEIGHBOR (NWKNN)

(STUDI KASUS: JURUSAN ILMU KEOLAHRAGAAN FAKULTAS ILMU KEOLAHRAGAAN UNIVERSITAS NEGERI MEDAN)

SKRIPSI

Diajukan sebagia persyaratan untuk memperoleh

gelar sarjana Komputer

Disusun Oleh:

Rizky Ramadhan

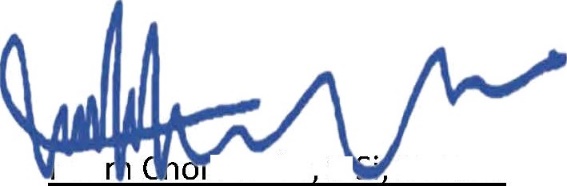
NIM: 155150200111059

Skripsi ini telah diuji dan dinyakan lulus pada

21 Desember 2020

Telah diperikasa dan disetuji oleh:

Dosen Pembimbing I Dosen Pembiming II



ma

issodin SJ M.Kom

Edy Santoso, S.Si, M.Kom

NIK: 201201 850719 1 001 NIK: 19740414 200312 1 004

Mengetahui

Ketua Jurusan Teknik Informatika

Achmad Basuki, S.T, M.MG, Ph.D

NIK: 19741118 200312 1 002

PERNYATAAN ORISINALITAS

Saya, menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak tedapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar referensi.

Apabila ternyata di dalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Medan, 21 Desember 2020

Rizky Ramadhan

155150200111059

KATA PENGANTAR

Assalamu’alaikum Warahmatullahi Wabarakatuh.

Syukur Alhamdulillah penulis sampaikan atas ke hadirat Allah Subhanahu Wa Ta’ala yang telah memberikan rahmat dan hidayah-Nya, Shalawat dan salam penulis kirimkan kepada Nabi Muhammad Shalallahu Alaihi Wasallam, beserta para sahabat dan keluarga beliau, sehingga penulis dapat menyelesaikan penelitian yang berjudul “REKOMENDASI PENGAMBILAN TEMA JUDUL SKRIPSI MENGGUNAKAN PSO-NEIGHBOR WEIGHTED K-NEAREST NEIGHBOR (NWKNN), (STUDI KASUS: JURUSAN ILMU KEOLAHRAGAAN FAKULTAS ILMU KEOLAHRAGAAN UNIVERSITAS NEGERI MEDAN)”. Dalam kesempatan ini penulis ingin menyampaikan rasa terima kasih kepada pihak-pihak yang telah membimbing dan membantu penulisa dalam penyusunan laporan penelitian ini, antara lain:

1. Bapak Imam Cholissodin, S.Si, M.Kom, selaku Dosen Pembimbing I yang telah baik hati dan sabar membimbing penulis dalam menyelesaikan penelitian ini.
2. Bapak Edy Santoso, S.Si, M.Kom, selaku Dosen pembimbing II yang telah dengan ikhlas membina dan mengarahkan penulis dalam menyelesaikan penyusunan penelitian ini.
3. Bapak Bayu Priyambadha, S.Kom, M.Kom, selaku Ketua Program Studi Teknik Informatika, Fakultas Ilmu Komputer Universitas Brawijaya.
4. Bapak Achmad Basuki, S.T, M.MG, Ph.D, selaku Ketua Jurusan Teknik Informatika, Fakultas Ilmu Komputer Universitas Brawijaya.
5. Bapak Adhitya Bhawiyuga, S.Kom, M.Sc, selaku Dosen Pembimbing Akademik yang telah membimbing penulis selama menempuh perkuliahan.
6. Keluarga penulis, terutama untuk Ibu dan Bapak yang telah memberikan semangat yang tidak hentinya beserta dukungan adik, paman dan bibi bagi penulis untuk menyelesaikan penelitian ini.
7. Thareq Ibrahim, Dary Farhan, M. Ainun Najib, Vasha Farishi, M. Dearifaldi selaku teman penulis yang telah memberikan banyak bantuan dalam proses perkuliahan dan pengerjaan penelitian.
8. Seluruh pihak yang telah mendukung dan membantu penulis dalam proses perkuliahan dan pengerjaan penelitian.

Penulis menyadari penelitian ini masih memiliki kekurangan, untuk itu penulis sangat mengharapkan kritik dan saran yang bersifat membangun. Semoga penelitian ini dapat digunakan dengan baik dan bermanfaat dalam membantu penelitian selanjutnya.

Medan, 21 Desember 2020

Penulis

Rizkyyud27@gmail.com

ABSTRAK

**Rizky Ramadhan, REKOMENDASI PENGAMBILAN TEMA JUDUL SKRIPSI MENGGUNAKAN *PSO-NEIGHBOR WEIGHTED K-NEAREST NEIGHBOR (NWKNN)***

**(STUDI KASUS: JURUSAN ILMU KEOLAHRAGAAN FAKULTAS ILMU KEOLAHRAGAAN UNIVERSITAS NEGERI MEDAN)**

**Pembimbing: Imam Cholissodin, S.Si, M.Kom dan Edy Santoso S.Si, M.Kom**

Dewasa ini dalam keadaan kondisi epidemi yang mewabah tahun 2020 ini membuat mahasiswa menjadi kurangnya tatap muka langsung dan berdiskusi dengan dosen soal materi skripsi yang ingin diangkat, dengan adanya kendaala ini mahasiswa menjadi sulit untuk mementukan skripsi apa yang ingin ia buat. Faktor utama kendala penulisan skripsi adalah “Menentukan Judul. Teknik K-Means sebagai pembentukan awal cluster judul skripsi mahasiswa yang telah tamat, Teknik PSO sebagai seleksi matakuliah, Teknik Neighbor Weighted K-Nearest Neighbor (NWKNN) sebagai klasifikasi data dan pengukuran performa algoritma dengan teknik akurasi. Pada pengujian persentase banyak perbandingan data Training dan data Testing yang digunakan sebesar 90%: 10%. Pada pengujian generasi dan fitur menghasilkan generasi yang mulai konstan pada generasi ke 50 dan dengan mata kuliah yang digunakan sebanyak 15 mata kuliah yaitu : MK2, MK6, MK7, MK11, MK12, MK13, MK14, MK15, MK21, MK24, MK26, MK27, MK28, MK31, MK32. Pada pengujian nilai K diperoleh nilai K optimum sebesar 3. Pada pengujian nilai K dan E diperoleh nilai K dan E optimum sebesar 3 dan 2. Algoritma PSO-Neighbor Weighted K-Nearest Neighbor (NWKNN) untuk rekomendasi pengambilan tema judul skripsi menghasilkan nilai optimum menggunakan parameter sebelumnya diperoleh hasil akurasi sebesar 88.28%.

Kata Kunci: Rekomendasi, Tema Judul Skripsi, *PSO*, Seleksi fitur, Klasifikasi, *Neighbor Weighted K-Nearest Neighbor* (NWKNN).

ABSTRACT

**Rizky Ramadhan, *RECOMMENDATION ON TITLE THEME TITLE USING PSO NEIGHBOR WEIGHTED K-NEAREST NEIGHBOR (NWKNN)***

***(CASE STUDY: DEPARTMENT OF SCHOOL OF SCIENCE FACULTY OF SCHOOL OF SCIENCE, MEDAN STATE UNIVERSITY)***

***Supervisors*: Imam Cholissodin, S.Si, M.Kom and Edy Santoso S.Si, M.Kom**

Nowadays, in an epidemic condition that is endemic in 2020, it makes students lack face-to-face and discuss with lecturers about the thesis material that they want to be appointed, with this condition students become difficult to determine what thesis they want to make. The main factor of thesis writing constraints is "Determining the Title. The K-Means technique is the initial formation of the clusters of student thesis titles, PSO technique as a course selection, the Neighbor Weighted K-Nearest Neighbor (NWKNN) Technique as data classification and algorithm performance measurement using accuracy techniques In testing the percentage of many comparisons of Training data and Testing data used is 90%: 10%. The generation and feature testing resulted in a generation that began to be constant in the 50th generation and with 15 subjects, namely: MK2, MK6, MK7, MK11, MK12, MK13, MK14, MK15, MK21, MK24, MK26, MK27, MK28, MK31, MK32. In testing the K value, the optimum K value is 3.In testing the K and E values, the optimum K and E values ​​are 3 and 2. The PSO-Neighbor Weighted K-Nearest Neighbor (NWKNN) algorithm for recommendations for the theme of the thesis title produces the optimum value using The previous parameter obtained an accuracy of 88.28%.

Keywords: *Recommendation, Thesis Title Theme, PSO, Feature Selection, Neighbor Weighted K-Nearest Neighbor (NWKNN) Classification.*

DAFTAR ISI

[PENGESAHAN ii](#_Toc59433452)

[PERNYATAAN ORISINALITAS iii](#_Toc59433453)

[KATA PENGANTAR iv](#_Toc59433454)

[ABSTRAK vi](#_Toc59433455)

[ABSTRACT vii](#_Toc59433456)

[DAFTAR ISI viii](#_Toc59433457)

[DAFTAR TABEL xi](#_Toc59433458)

[DAFTAR GAMBAR xiii](#_Toc59433459)

[DAFTAR KODE PROGRAM xiv](#_Toc59433460)

[DAFTAR LAMPIRAN xv](#_Toc59433461)

[BAB 1 PENDAHULUAN 16](#_Toc59433462)

[1.1 Latar Belakang 16](#_Toc59433463)

[1.2 Rumusan Masalah 17](#_Toc59433464)

[1.3 Tujuan 17](#_Toc59433465)

[1.4 Manfaat 18](#_Toc59433466)

[1.5 Batasan Masalah 18](#_Toc59433467)

[1.6 Sistematika Pembahasan 18](#_Toc59433468)

[BAB 2 LANDASAN KEPUSTAKAAN 20](#_Toc59433469)

[2.1 Kajian Pustaka 20](#_Toc59433470)

[2.2 Skripsi 23](#_Toc59433471)

[2.3 Data *Mining* 23](#_Toc59433472)

[2.4 K-Means 23](#_Toc59433473)

[2.5 Algoritma Seleksi Atribut *Particle Swarm Optimization* (PSO) 24](#_Toc59433474)

[2.5.1 Inisialisasi Awal 24](#_Toc59433475)

[2.5.2 *Update* Kecepatan 24](#_Toc59433476)

[*2.5.3 Update* Posisi 25](#_Toc59433477)

[2.5.4 *Update* Pbest dan Gbest 25](#_Toc59433478)

[2.5.5 Kondisi Berhenti 25](#_Toc59433479)

[2.6 Algoritma *Neighbor Weight K-Nearest Neighbor* (NWKNN) 25](#_Toc59433480)

[2.6.1 Euclidean Distance 26](#_Toc59433481)

[2.6.2 Perhitungan Bobot 26](#_Toc59433482)

[2.6.3 Perhitungan Skor 27](#_Toc59433483)

[2.7 Normalisasi Data 27](#_Toc59433484)

[2.8 Pengukuran Performa Algoritma 27](#_Toc59433485)

[BAB 3 METODOLOGI 28](#_Toc59433486)

[3.1 Tipe Penelitian 28](#_Toc59433487)

[3.2 Strategi Penelitian 28](#_Toc59433488)

[3.3 Lokasi Penelitian 29](#_Toc59433489)

[3.4 Teknik Pengumpulan Data 30](#_Toc59433490)

[3.4.1 Teknik Analisis Data 30](#_Toc59433491)

[3.4.2 Penerapan Algoritma 30](#_Toc59433492)

[3.5 Peralatan Pendukung 30](#_Toc59433493)

[3.5.1 Perangkat Keras (*Hardware*) 30](#_Toc59433494)

[3.5.2 Perangkat Lunak (*Software*) 31](#_Toc59433495)

[BAB 4 PERANCANGAN 32](#_Toc59433496)

[4.1 Diagram Alir Perancangan Algoritma 32](#_Toc59433497)

[4.1.1 Diagram Alir *Pre-processing* Data 34](#_Toc59433498)

[4.1.2 Diagram Alir Normalisasi Data 36](#_Toc59433499)

[4.1.3 Diagram Alir Penyeleksian Fitur 39](#_Toc59433500)

[4.1.4 Diagram Alir Perhitungan Jarak 41](#_Toc59433501)

[4.1.5 Diagram Alir Perhitungan Bobot 42](#_Toc59433502)

[4.1.6 Diagram Alir Perhitungan Skor 44](#_Toc59433503)

[4.2 Perhitungan Manualisasi 47](#_Toc59433504)

[4.2.1 Inisialisasi Data Numerik, Data *Training* dan Data *Testing* 49](#_Toc59433505)

[4.2.2 Perhitungan Normalisasi 53](#_Toc59433506)

[4.2.3 Pengklasteran Data 56](#_Toc59433507)

[4.2.4 Penyeleksian Fitur 57](#_Toc59433508)

[4.2.5 Perhitungan Jarak 61](#_Toc59433509)

[4.2.6 Pengurutan Jarak 62](#_Toc59433510)

[4.2.7 Perhitungan Bobot 62](#_Toc59433511)

[4.2.8 Perhitungan Skor 63](#_Toc59433512)

[4.2.9 Penghitungan Performa 64](#_Toc59433513)

[4.3 Perancangan Pengujian Algoritme 64](#_Toc59433514)

[4.3.1 Pengujian Persentase Banyak Data *Training* dan Data *Testing* 64](#_Toc59433515)

[4.3.2 Pengujian Banyak Generasi dan Fitur 65](#_Toc59433516)

[4.3.3 Pengujian Banyak Nilai K 66](#_Toc59433517)

[4.3.4 Pengujian Pengaruh Nilai *K* dan *E* 67](#_Toc59433518)

[BAB 5 IMPLEMENTASI 68](#_Toc59433519)

[5.1 Implementasi Sistem 68](#_Toc59433520)

[5.1.1 Implementasi *Pre-Processing Data* 68](#_Toc59433521)

[5.1.2 Implementasi Normalisasi data 69](#_Toc59433522)

[5.1.3 Implementasi Penyeleksian Fitur (PSO) 70](#_Toc59433523)

[5.1.4 Implementasi Perhitungan Jarak 71](#_Toc59433524)

[5.1.5 Implementasi Perhitungan Bobot 73](#_Toc59433525)

[5.1.6 Implementasi Perhitungan Skor 74](#_Toc59433526)

[BAB 6 HASIL DAN PEMBAHASAN 77](#_Toc59433527)

[6.1 Hasil Pengujian 77](#_Toc59433528)

[6.1.1 Pengujian banyak data 77](#_Toc59433529)

[6.1.2 Pengujian Banyak Generasi dan Fitur 78](#_Toc59433530)

[6.1.3 Pengujian Nilai K 80](#_Toc59433531)

[6.1.4 Pengujian Pengaruh Nilai K dan E 81](#_Toc59433532)

[BAB 7 PENUTUP 82](#_Toc59433533)

[7.1 Kesimpulan 82](#_Toc59433534)

[7.2 Saran 82](#_Toc59433535)

[DAFTAR REFERENSI 83](#_Toc59433536)

DAFTAR TABEL

[Tabel 2.1 Daftar Kajian Pustaka 21](#_Toc59433537)

[Tabel 4.1 Data Nilai Mata kuliah Mahasiswa dan Judul Skripsi 47](#_Toc59433538)

[Tabel 4.2 47](#_Toc59433539)

[Tabel 4.3 Konversi Nilai Huruf 49](#_Toc59433540)

[Tabel 4.4 Hasil Konversi Nilai Mata kuliah Mahasiswa 49](#_Toc59433541)

[Tabel 4.5 Data *Training* 51](#_Toc59433542)

[Tabel 4.6 Data *Testing* 52](#_Toc59433543)

[Tabel 4.7 Nilai Minimum dan Maksimum 53](#_Toc59433544)

[Tabel 4.8 Normalisasi Data *Training* 54](#_Toc59433545)

[Tabel 4.9 Normalisasi Data *Testing* 55](#_Toc59433546)

[Tabel 4.10 Pemberian Label Pada Data Latih 56](#_Toc59433547)

[Tabel 4.11 Inisialisasi Awal Partikel 58](#_Toc59433548)

[Tabel 4.12 Pbest Awal 58](#_Toc59433549)

[Tabel 4.13 Gbest Awal 58](#_Toc59433550)

[Tabel 4.14 Hasil *Update* Kecepatan 59](#_Toc59433551)

[Tabel 4.15 Sigmoid 59](#_Toc59433552)

[Tabel 4.16 Nilai Random 60](#_Toc59433553)

[Tabel 4.17 *Update* Posisi (t=1) 60](#_Toc59433554)

[Tabel 4.18 *Update* *Pbest* 60](#_Toc59433555)

[Tabel 4.19 Update *Gbest* 60](#_Toc59433556)

[Tabel 4.20 Hasil Terbaik 61](#_Toc59433557)

[Tabel 4.21 Hasil Jarak Data Uji 1 61](#_Toc59433558)

[Tabel 4.22 Pengurutan Jarak 62](#_Toc59433559)

[Tabel 4.23 *Weight* Kelas 63](#_Toc59433560)

[Tabel 4.24 Skor Kelas 63](#_Toc59433561)

[Tabel 4.25 Perbandingan Hasil Kelas 64](#_Toc59433562)

[Tabel 4.26 Perancangan Pengujian Presentasi Banyak Data *Training* dan Data *Testing* 65](#_Toc59433563)

[Tabel 4.27 Perancangan Pengujian Banyak Generasi Dan Fitur 65](#_Toc59433564)

[Tabel 4.28 Perancangan Pengujian bayak Nilai *K* 66](#_Toc59433565)

[Tabel 6.1 Hasil Pengujian Persentase Banyak Data *Training* dan Data *Testing* 77](#_Toc59433566)

[Tabel 6.2 Hasil Pengujian Banyak Generasi dan Fitur 78](#_Toc59433567)

[Tabel 6.3 Hasil Pengujian Nilai K 80](#_Toc59433568)

[Tabel 6.4 Hasil Pengujian Pengaruh Nilai K dan E 81](#_Toc59433569)

DAFTAR GAMBAR

[Gambar 3.1 Gambar Diagram Blok Prediksi Tema Judul Skripsi dengan Algoritma Neighbor Weight K-Nearest Neighbor (NWKNN) 29](file:///C:\Users\ASUS\OneDrive\Desktop\Skripsi\Final_Skripsi_RizkyRamadhan_S2_155150200111059.docx#_Toc59465490)

[Gambar 4.1 Diagram Alir Perancangan Algoritma 33](file:///C:\Users\ASUS\OneDrive\Desktop\Skripsi\Final_Skripsi_RizkyRamadhan_S2_155150200111059.docx#_Toc59465491)

[Gambar 4.2 Diagram Alir Pre-processing Data 35](file:///C:\Users\ASUS\OneDrive\Desktop\Skripsi\Final_Skripsi_RizkyRamadhan_S2_155150200111059.docx#_Toc59465492)

[Gambar 4.3 Diagram Alir Normalisasi 37](file:///C:\Users\ASUS\OneDrive\Desktop\Skripsi\Final_Skripsi_RizkyRamadhan_S2_155150200111059.docx#_Toc59465493)

[Gambar 4.5 Diagram Alir Seleksi Fitur 40](file:///C:\Users\ASUS\OneDrive\Desktop\Skripsi\Final_Skripsi_RizkyRamadhan_S2_155150200111059.docx#_Toc59465494)

[Gambar 4.6 Diagram Alir Perhitungan Jarak 41](file:///C:\Users\ASUS\OneDrive\Desktop\Skripsi\Final_Skripsi_RizkyRamadhan_S2_155150200111059.docx#_Toc59465495)

[Gambar 4.7 Diagram Alir Pembobotan 43](file:///C:\Users\ASUS\OneDrive\Desktop\Skripsi\Final_Skripsi_RizkyRamadhan_S2_155150200111059.docx#_Toc59465496)

[Gambar 4.8 Diagram Alir Penentuan Skor 46](file:///C:\Users\ASUS\OneDrive\Desktop\Skripsi\Final_Skripsi_RizkyRamadhan_S2_155150200111059.docx#_Toc59465497)

[Gambar 6.1 Hasil Pengujian Persentase Banyak Data *Training* dan Data *Testing* 78](#_Toc59465498)

[Gambar 6.2 Hasil Pengujian Generasi dan Fitur 79](file:///C:\Users\ASUS\OneDrive\Desktop\Skripsi\Final_Skripsi_RizkyRamadhan_S2_155150200111059.docx#_Toc59465499)

[Gambar 6.3 Hasil Pengujian Nilai K 80](#_Toc59465500)

[Gambar 6.4 Hasil Pengujian Pengaruh NIlai K dan E 81](file:///C:\Users\ASUS\OneDrive\Desktop\Skripsi\Final_Skripsi_RizkyRamadhan_S2_155150200111059.docx#_Toc59465501)

DAFTAR KODE PROGRAM

[Kode Program 5.2 *Pre-Processing* Data 68](#_Toc59379357)

[Kode Program 5.3 Normalisasi Data 70](#_Toc59379358)

[Kode Program 5.4 Penyeleksian Fitur (PSO) 71](#_Toc59379359)

[Kode Program 5.5 Perhitungan Jarak 72](#_Toc59379360)

[Kode Program 5.6 Perhitungan Bobot 74](#_Toc59379361)

[Kode Program 5.7 Perhitungan Skor 75](#_Toc59379362)

DAFTAR LAMPIRAN

[LAMPIRAN A DATA 85](#_Toc59388027)

[LAMPIRAN B Daftar mata kuliah 86](#_Toc59388028)

# PENDAHULUAN

Pada bagian ini menjelaskan permasalahan yang dihadapi sehingga dapat di angkat menjadi penelitian, permasalahan yang dihadapi sebagai berikut:

## Latar Belakang

Dewasa ini dalam keadaan kondisi epidemi yang mewabah tahun 2020 ini membuat mahasiswa menjadi kurangnya tatap muka langsung dan berdiskusi dengan dosen soal materi skripsi yang ingin diangkat, untuk memaksimalkan pertemuan dan pembahasan persoalan skripsi mahasiswa diharapkan dapat mengetahui gambaran dasar yang ingin diangkat. Sehingga nantinya dosen pembimbing dapat memberikan konsultasi yang lebih baik terhadap persoalan yang ingin diangkat, namun mahasiswa banyak yang masih bingung dan bimbang dalam permasalahan skripsi.

Berdasarkan penelitian sebelumnya beberapa faktor kendala mahasiswa terhadap skripsi berkaitan dengan teknik penyusunan laporan skripsi, terdapat  faktor yang mempengaruhi penulisan skripsi diantaranya; “Menentukan Judul”, ”Menemukan Masalah”, “Proses Analisis Data”, “Menulis Laporan Penelitian”, “Menulis Proposal”, “Mendapatkan Teman Diskusi”, “Pengambilan Data Penelitian”, “Mendapat Izin dari Lokasi Penelitian”, “Menemui Pembimbing”, “Mendapatkan Referensi yang Relevan” (Widarto, 2017).

Faktor utama kendala penulisan skripsi adalah “Menentukan Judul” dengan tingkat kategori yang “Tinggi” (Widarto, 2017), untuk menentukan suatu judul mahasiswa harus mengetahui mata kuliah yang terkait dengan judul tersebut, terdapat sebuah pengaruh sedang atau cukup antara mata kuliah yang diambil mahasiswa terhadap kemampuan pengerjaan skripsi mahasiswa. Pengaruh mata kuliah tersebut seringkali diabaikan oleh para mahasiswa sehingga menyebabkan sulitnya mahasiswa untuk menentukan judul yang ingin diajukan kepada dosen pembimbing dan serta pemahaman untuk menjelaskan judul tersebut kepada dosen pembimbingnya, keresahan dan kegalauan tersebut sering kali menjadi pikiran mahasiswa yang tak usai sehingga mereka tidak berani untuk melanjutkan dan menjalankan skripsinya dengan serius.

Sebelum memberikan rekomendasi judul pada mahasiswa terdapat berbagai macam jenis-jenis judul sehingga dibutuhkan pengkategorian terhadap judul skripsi mahasiswa tersebut sehingga lebih memudahkan memberikan rekomendasi judul mahasiswa tersebut berdasarkan kategori yang telah dibuat. Untuk pembuatan kategori dibutuhkan algoritma *clustering* sehingga dapat menghasilkan kategori yang dibutuhkan untuk klasifikasi selanjutnya.

Dalam penelitian sebelumnya (Somantri, et al., 2016), dilakukan penelitian untuk membuat klasifikasi tema tugas akhir mahasiswa menggunakan kombinasi teknik *K-Means* sebagai pembentukan cluster dan *Support Vector Machine* (SVM) sebagai teknik klasifikasi. Berdasarkan penelitian tersebut peneliti mengambil teknik K-Means sebagai pembentukan awal cluster judul skripsi mahasiswa yang telah tamat.

Selanjutnya permasalahan yang berkaitan dengan mata kuliah yang digunakan untuk bahan perhitungan klasifikasi, untuk mata kuliah yang digunakan yaitu mata kuliah wajib fakultas dan mata kuliah keminatan. Namun mata kuliah yang digunakan tersebut diseleksi keterkaitannya dengan cluster yang telah dibuat sebelumnya menggunakan metode seleksi atribut. Pada penelitian (Hasfi, et al., 2016) terdapat peningkatan akurasi sehingga menghasilkan akurasi maksimal sebesar 83,6%.Untuk metode pengklasifikasian dibutuhkan sebuah metode yang dapat mencocokan antara data latih dengan *dataset* dikarenakan untuk konsep pemberian rekomendasi tema judul skripsi melihat kedekatan/kemiripan nilai dari nilai mahasiswa yang ingin mencari judul dengan nilai-nilai mahasiswa yang telah tamat. Metode yang digunakan merupakan metode *Nearest Neighbor* (NN) berupa metode *lazy learner* yaitu sebuah metode klasifikasi yang dilakukan ketika ada data uji yang ingin diketahui label kelasnya, maka ketika data uji tersedia algoritma tersebut melakukan klasifikasi berdasarkan kemiripan suatu data dengan data yang lain (Prasetyo, 2014).

Algoritma Nearest Neighbor (NN) yang digunakan merupakan algoritma *Neighbor Weight K-Nearest Neighbor* (NWKNN), alasan pengambilan algoritma ini karena memiliki akurasi yang lebih baik dibandingkan algoritma K-Nearest Neighbor (KNN) berdasarkan penelitian (Hadi, et al., 2018) dengan hasil akurasi NWKNN rata-rata lebih baik 5% dibandingkan dengan hasil metode KNN dari hasil setiap pengujian. Dengan adanya penelitian ini diharapkan dapat mengurangi dan membantu kegundahan mahasiswa dalam penentuan tema yang diangkat menjadi judul skripsi yang ingin dia kerjakan. Sehingga nantinya mahasiswa dapat bertemu dengan dosen sudah memiliki tema yang dikonsultasikan kepada dosen pembimbing dan menambah efisiensi waktu bimbingan mahasiswa.

## Rumusan Masalah

Rumusan Masalah dari penelitian ini sebagai berikut:

1. Bagaimana nilai parameter yang tepat pada penerapan algoritma NWKNN untuk memberikan rekomendasi terbaik terkait tema judul skripsi kepada mahasiswa?
2. Bagaimana nilai akurasi terbaik dari penerapan algoritma NWKNN untuk memberikan rekomendasi terbaik terkait tema yang diangkat menjadi judul skripsi mahasiswa?

## Tujuan

Tujuan dari penelitian ini sebagai berikut:

1. Mengetahui nilai parameter yang tepat pada penerapan algoritma NWKNN untuk memberikan rekomendasi terbaik terkait tema yang diangkat menjadi judul skripsi mahasiswa.
2. Mengetahui nilai akurasi terbaik dari penerapan algoritma NWKNN untuk memberikan rekomendasi terbaik terkait tema yang diangkat menjadi judul skripsi mahasiswa.

## Manfaat

Manfaat penelitian ini sebagai berikut:

1. Menambah wawasan dan kemampuan berpikir dalam penerapan algoritma NWKNN dalam penyelesaian permasalahan.
2. Menambah pengetahuan untuk melihat keterkaitan mata kuliah dan kemampuan mahasiswa dalam menyelesaikan skripsi.
3. Memberikan rekomendasi tema skripsi kepada mahasiswa sehingga dapat membantu dalam mengembangkan judul.

## Batasan Masalah

Batasan masalah yang ada pada penelitian ini sebagai berikut:

1. Mata kuliah yang diambil sebagai bahan perhitungan merupakan mata kuliah wajib fakultas dan mata kuliah peminatan ilmu keolahragaan (IKOR).
2. Data Mahasiswa yang digunakan menjadi data latih merupakan data mahasiswa wisudawan tahun 2016-2019 sebanyak 180 data.
3. Pengujian hasil keakuratan hasil prediksi menggunakan teknik akurasi.

## Sistematika Pembahasan

Untuk memahami lebih jelas penelitian ini, maka materi-materi dikelompokan menjadi beberapa sub bagian sebagai berikut:

**BAB I PENDAHULUAN**

Berisi tentang latar belakang, perumusan masalah, tujuan, manfaat batasan masalah dan sistematika pembahasan.

**BAB II LANDASAN KEPUSTAKAAN**

Bab ini berisikan teori yang berupa pengertian dan definisi yang diambil dari kutipan sumber-sumber yang berkaitan dengan penyusunan laporan skripsi yang berhubungan dengan penelitian.

**BAB III METODOLOGI**

Bab ini menjelaskan metode yang digunakan dalam penelitian yang berisi gambaran pengambilan data, pengolahan data dan penggunaan algoritma pada data yang digunakan pada penelitian ini.

**BAB IV PERANCANGAN**

Pada bagian ini akan membahas mengenai perancangan dari hasil ini akan dipaparkan mengenai formulasi permasalahan, desain arsitektur sistem, 5 perancangan algoritma dari mulai diagram alir dari proses *training*, *testing*, *evaluasi*, perhitungan manualisasi dan skenario pengujian yang digunakan*.*

# LANDASAN KEPUSTAKAAN

Pada bagian ini menjelaskan sumber-sumber dan bahan-bahan yang digunakan dalam menjalankan penelitian, landasan yang digunakan untuk penelitian sebagai berikut.

## Kajian Pustaka

Sebagai bahan rujukan peneliti terdapat beberapa bahan penelitian yang telah lalu. Pada kajian pustaka awal sebagai bahan rujukan mengangkat judul skripsi sebagai pokok utama pembahasan merujuk pada penelitian yang telah dilakukan oleh Widarto (2017) dalam penelitian Faktor Penghambat Studi Mahasiswa yang Tidak Lulus Tepat Waktu Di Jurusan Pendidikan Teknik Mesin FT UNY yang dilakukan pada mahasiswa D3 dan S1 dengan metode pendekatan berupa survey. Dan untuk mahasiswa S1 yang sebagai object utama yang diangkat pada penelitian ini dibagi 3 kelompok masalah mahasiswa berupa posisi skripsi, bimbingan skripsi frekuensi, bimbingan diharapkan. Pada kasus kelompok posisi skripsi pengajuan judul menempati posisi teratas dengan indeks kesulitan tertinggi, sehingga menjadi bahan rekomendasi peneliti menjadi dasar masalah pada penelitian ini.

Pada penentuan penggunaan *K-Means* sebagai teknik pengklasteran judul skripsi merujuk pada penelitian sebelumnya yang telah dilakukan oleh Oman Somantri, Slamet Wiyono, dan Dairoh (2016) dalam melakukan Metode *K-Means* Untuk Optimasi Klasifikasi Tema Tugas Akhir Mahasiswa Menggunakan *Support Vector Machine* (SVM). Pada skema pengujian ketepatan hasil menggunakan teknik akurasi dengan total data latih 131 mahasiswa. Pada hasil penelitian tersebut dapat perbandingan akurasi antara hasil klasifikasi SVM dan klasifikasi SVM dengan *K-Means* dimana hasil tingkat akurasi sebelumnya 85.38% dan 86.21%. Pada penentuan penggunaan *Particle Swarm Optimization* sebagai seleksi fitur pada mata kuliah yang digunakan merujuk pada penelitian sebelumnya yang telah dilakukan oleh (Hasfi, et al., 2016) tentang “Klasifikasi Kinerja Akademik Mahasiswa Menggunakan Metode PSO - SVM” yang diharapkan dapat membantu pihak Fakultas Ilmu Komputer Universitas Brawijaya untuk mengetahui perkembangan mahasiswa tiap semesternya dan dapat mengevaluasi kinerja akademik mahasiswa secara dini dengan menghasilkan akurasi 83,6%.

Pada kajian pustaka selanjutnya untuk mencari teknik *lazy learner* yang digunakan rujukan dari beberapa penelitian diantaranya. Penelitian sebelumnya yang telah dilakukan oleh Luthfi Anshori, Rekyan Regasari Mardi Putri, Tibyani (2018) dalam melakukan Implementasi Metode KNN untuk rekomendasi peminatan studi pada mahasiswa jurusan Teknik Informatika Universitas Brawijaya Malang. Data dalam penelitian ini menggunakan data mahasiswa berjumlah 185 dengan akurasi 76.66%. Pada rujukan selanjutnya merupakan penelitian yang dilakukan oleh Azizul Hanifah Hadi, Dian Eka Ratnawati, Candra Dewi (2018) untuk mengidentifikasi penyakit ginjal menggunakan metode NWKNN dengan data sebanyak 150 data latih dan 50 data uji. Pada hasil penelitian tersebut diketahui bahwa tingkat akurasi yang dihasilkan metode NWKNN memiliki selisih nilai sebesar 5% dibandingkan metode KNN. Metode NWKNN dengan akurasi 70% dan metode KNN dengan akurasi 65%.

Pada penelitian sebelumnya yang dilakukan oleh Indriati dan Achmad Ridok untuk melakukan sentiment analysis untuk pemberian ulasan dengan menggunakan metode NWKNN dengan menggunakan data *review* sebanyak 550 data dengan keterangan 500 data review negative dan 50 data *review* positif. Pada hasil penelitian tersebut menghasilkan tingkat F-Measure tertinggi pada KNN yaitu 0,522 dan pada NWKNN yaitu 0,797 dengan perbedaan NWKNN lebih baik 0,27 dibandingkan KNN. Pada penelitian lain yang dilakukan oleh Aldy Satria, Marji, Dian Eka Ratnawati untuk melakukan klasifikasi jenis kanker berdasarkan struktur protein dengan metode NWKNN dengan menggunakan data uji sebanyak 602 data latih dan 105 data uji menghasilkan tingkat akurasi tertinggi NWKNN sebesar 80,66% dan KNN sebesar 80,00%. Pada penelitian (Akromunnisa & Hidayat, 2019) dilakukan pengujian *Holdout validation* dan *K-fold cross validation* dalam klasifikasi dokumen tugas akhir semester (skripsi) menggunakan *K-Nearest Neighbor*.

Tabel 2.1 Daftar Kajian Pustaka

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Penelitian | Objek | Metode | Hasil |
| 1 | (Widarto, 2017) | Mahasiswa jenjang pendidikan d3, s1 teknik mesin ft uny | Survey | Tabel ranking kesulitan permasalahan penyusunan tugas akhir bagi d3 dan skripsi bagi s1 |
| 2 | (Somantri, et al., 2016) | Tema judul tugas akhir mahasiswa | *K-means* dengan  3 cluster dan teknik klasifikasi *support vector machine* (svm) | Hasil akurasi svm tanpa menggunakan k-means 85,38% dan svm dengan k-means 86,21% |
| 3 | (Hasfi, et al., 2016) | Klasifikasi Kinerja Akademik Mahasiswa Menggunakan Metode PSO-SVM | Seleksi fitur menggunakan PSO dan SVM sebagai teknik Klasifikasi | metode PSO-SVM memiliki tingkat akurasi yang baik yaitu sebesar 83,6% dengan fitur F3, F8, F9, F13, F14, F17, F21, dan F24, jumlah iterasi *PSO* sebanyak 100 iterasi, jumlah partikel sebanyak 7 partikel, nilai *λ* dengan interval [8, 10], nilai *C* dengan interval [40, 50], nilai *γ* dengan interval [0.5, 1], jumlah iterasi *SVM* sebanyak 1000 iterasi, dan *ε* sebesar 0.01. |
| 4 | (Anshori, et al., 2018) | Peminatan mahasiswa s1 jurusan teknik informatika | K-nearest neighbor | Hasil akurasi K-NN terbaik yang didapat bernilai 76,66% dengan nilai k=10 |
| 5 | (Hadi, et al., 2018) | Identifikasi penyakit gagal ginjal | Neighbor Weight K-Nearest Neighbor (NKWNN) | Hasil akurasi yang terbaik didapat oleh NWKNN bernilai 70% dan KNN bernilai 65% dengan K=2 |
| 6 | (Indriati & Ridok, 2016) | *Sentiment analysis* untuk ulasan aplikasi *mobile* | Neighbor Weighted K-Nearest Neighbor (NWKNN) | Hasil f-measure terbesar pada KNN yaitu 0,522 dan pada NWKNN yaitu 0,797 |
| 7 | (Satria, et al., 2019) | Jenis kanker berdasarkan struktur protein | Neighbor Weighted K-Nearest Neighbor (NWKNN) | Hasil Akurasi dengan metode NWKNN menghasilkan akurasi sebesar 80,66% dan KNN menghasilkan akurasi 80,00% |
| 8 | (Akromunnisa & Hidayat, 2019) | Perbandingan akurasi Hold validation dan K-Fold validation pada metode knn | Klasifikasi dokumen tugas akhir (skripsi) menggunakan k-nearest neighbor | Menggunakan pembagian data Split into train test sets dengan rasio perbandingan 9:1 menghasilkan akurasi lebih besar dibandingkan dengan rasio perbandingan 6:4, 7:3, 8:2 dan pembagian data menggunakan kfold cross validation. |

Berdasarkan penelitian sebelumnya peneliti akhirnya mengangkat tema skripsi berkaitan dengan pencarian tema judul skripsi mahasiswa dengan teknik klasifikasi algoritma *Neighbor Weight K-Nearest Neighbor* (NWKNN) dan teknik seleksi fitur PSO sebagai metode yang digunakan untuk pemecahan permasalahan tersebut.

## Skripsi

Skripsi adalah karya ilmiah mahasiswa pendidikan program sarjana (S1) yang merupakan wujud dari kajian pengetahuan dan/atau penerapan teknologi berdasarkan kaidah ilmiah dalam minat studi yang telah dipelajarinya. Dengan demikian, sebuah karya ilmiah berupa skripsi merupakan hasil penelitian sederhana yang disusun dengan menggunakan metode ilmiah baku serta memenuhi unsur kesesuaian dengan bidang keahlian yang dipelajari mahasiswa pada salah satu Program Studi (Prodi) yang ditempuhnya.

Penentuan ukuran kesesuaian skripsi salah satunya dapat dikaji berdasarkan kompetensi yang harus dipunyai lulusan sebuah Program Studi. Filkom UB memiliki 5 (lima) Prodi S1, yakni: Teknik Informatika, Sistem Informasi, Pendidikan Teknologi Informasi, Teknik Komputer, dan Teknologi Informasi. Selanjutnya, profil lulusan, yang akan dihasilkan dari setiap Prodi, telah dinyatakan pada kurikulum berbasis SN-DIKTI dari masing-masing Prodi yang bersesuaian. Oleh karena itu, skripsi sebagai bentuk tugas akhir sarjana tentunya harus mencerminkan setidaknya satu dari profil lulusan tersebut.

Di samping itu, sebuah skripsi juga harus memenuhi kriteria-kriteria yang menyangkut kedalaman karya ilmiah pada tingkatan program sarjana (S1) yang disesuaikan dengan jenis/tipe penelitiannya. Untuk memastikan bahwa mahasiswa mampu memenuhi kedalaman skripsi tersebut, maka panduan ini mengarahkan mahasiswa untuk memiliki kemampuan membuat konsep, menganalisis, merencanakan atau merancang, mengimplementasikan, mengoperasikan, dan mengevaluasi sistem sesuai dengan permasalahan yang diangkat dalam penelitiannya (UB, 2018).

## Data *Mining*

Data mining, sering juga disebut Knowledge Discovery in Database (KDD), adalah kegiatan yang meliputi pengumpulan, pemakaian data historis untuk menemukan keteraturan, pola atau hubungan dalam set data berukuran besar. Keluaran dari data mining ini bisa dipakai untuk memperbaiki pengambilan keputusan di masa depan (Santosa, 2007).

## K-Means

K-Means ini digunakan untuk membuat kluster-kluster atau tema-tema dari judul skripsi mahasiswa yang nantinya hasil dari pengklusteran tersebut dijadikan sebagai kelas untuk proses klasifikasi menggunakan algoritma NWKNN. *K-Means Clustering* merupakan metode pengelompokan paling sederhana yang mengelompokkan data kedalam *k* kelompok berdasar pada *centroid* masing-masing kelompok. Hanya saja hasil dari *K-Means* sangat dipengaruhi parameter *k* dan inisialisasi *centroid*. Pada umumnya *K-Means* menginisialisasi *centroid* secara acak (Somantri, et al., 2016).

## Algoritma Seleksi Atribut *Particle Swarm Optimization* (PSO)

Algoritma Seleksi atribut ini digunakan untuk menyeleksi mata kuliah yang digunakan sehingga dapat meningkatkan hasil akurasi dan untuk mencari *fitness* digunakan algoritma NWKNN dengan nilai K = 5 dan nilai E = 2. Terdapat beberapa langkah-langkah untuk melakukan penyeleksian fitur berdasarkan (Cholissodin & Riyandani, 2016) diantaranya.

### Inisialisasi Awal

Inisialisasi dilakukan untuk membangkitkan himpunan solusi baru secara acak/*random* yang terdiri atas sejumlah *string* dimensi partikel dan ditempatkan pada penampungan yang disebut populasi. Dalam tahap ini harus ditentukan ukuran populasi (*popSize*). Nilai ini menyatakan banyaknya individu/partikel yang ditampung dalam populasi. Panjang setiap string dimensi partikel (*string In*) dihitung berdasarkan presisi variabel solusi yang kita cari.

### *Update* Kecepatan

*Update* Kecepatan dilakukan untuk menentukan arah perpindahan posisi partikel yang ada di populasi. Batasan *lower* dan *upper* kecepatan yang digunakan dalam proses ini berdasarkan nilai maksimum dari posisi partikel (*xmax*). Diketahui nilai *lower* dan *upper* untuk *xi*,*j*=[*xmin*, *xmax*]=[0,1], karena memang nilai *xi*,*j* tersebut biner, dan untuk *vi*,*j*=[*vmin*,*vmax*], di-mana *vmin*=-*vmax* dan *vmax*=60%\**xmax.* Dalam menentukan interval ini, baik untuk posisi maupun kecepatan partikel, sebaiknya dilakukan pada proses inisialisasi.

Maka untuk mendapatkan hasil *update* kecepatannya dihitung sebagai berikut dengan menggunakan Persamaan 2.1.

**(2.1)**

Keterangan:

: Kecepatan Baru

: Bobot inertia

: Kecepatan Sebelumnya

: Nilai Koefisien Akselerasi 1

: Nilai Koefisien Akselerasi 2

: Nilai Random [0, 1]

: Nilai Random [0, 1]

: Nilai Posisi Partikel pada dimensi Ke-j

: Nilai Pbest dari dimensi Ke-j

: Nilai Gbest dari dimensi Ke-j

### *Update* Posisi

Setelah menghitung kecepatan, kemudian hitung Sig(vi,j(t)). Berikut cara menghitungnya dengan pada Persamaan 2.2.

**(2.2)**

Keterangan:

: Sigmoid dari kecepatan indeks *i*, *j*

: Exp

Selanjutnya membuat nilai random [0, 1] untuk membandingkan dengan nilai sigmoid dengan rumus Persamaan 2.3.

**(2.3)**

Keterangan:

: Nilai Posisi Partikel Sesudahnya pada dimensi Ke-j

: Kecepatan Baru

: Sigmoid dari kecepatan indeks *i*, *j*

### *Update* Pbest dan Gbest

*Update* *Pbest*, disini kita harus membandingkan antara *Pbest* pada iterasi sebelumnya dengan hasil fitness dari *update* Posisi generasi terbaru dengan menggunakan Persamaan 2.4.

**(2.4)**

Dan untuk *update* Gbest membandingkan Gbest iterasi sebelumnya dengan nilai Pbest tertinggi pada generasi terbaru.

### Kondisi Berhenti

Untuk proses kondisi berhenti disesuaikan dengan jumlah generasi yang diinginkan sehingga jika pengguna menginginkan 100 generasi maka proses pencarian selesai hingga iterasi ke-100.

## Algoritma *Neighbor Weight K-Nearest Neighbor* (NWKNN)

Metode NWKNN merupakan pengembangan metode KNN, perbedaan terdapat pada pemberian bobot yang dilakukan algoritma NWKNN (Feizar, et al., 2014). Pada tetangga yang berasal dari kelas mayoritas maka diberi bobot yang kecil, sedangkan pada tetangga yang berasal dari kelas minoritas maka diberi bobot yang besar (Ridok & Latifah, 2015).

Persamaan matematika harus diberi nomor urut dalam kurung biasa dan harus diacu dalam tulisan. Langkah-langkah pada algoritma NWKNN adalah sebagai berikut (Fadila, et al., 2016):

1. Menentukan nilai K tetangga.

2. Menghitung nilai kedekatan ketetanggaan antara data uji dan data latih menggunakan *Euclidean Distance*.

3. Mengurutkan hasil perhitungan kedekatan ketetanggaan berdasar jarak atau kedekatan dalam kelompok/*similarity*.

4. Mengumpulkan kategori klasifikasi *nearest neighbor*.

5. Perhitungan bobot.

6. Perhitungan Skor.

### Euclidean Distance

Untuk menghitung nilai kedekatan ketetanggaan menggunakan rumus Euclidean Distance dapat menggunakan Persamaan 2.5.

**(2.5)**

Keterangan:

: Nilai tiap data latih

: Nilai tiap data uji

: Banyak fitur

: Indeks fitur ke-

### Perhitungan Bobot

Untuk melakukan pembobotan digunakan rumus bobot (Tan, 2005) menggunakan Persamaan 2.6.

**(2.6)**

Keterangan:

: Banyak data latih pada kelas

: Banyak data latih pada kelas dimana terdapat dalam himpunan tetangga terdekat

: Nilai *exponen* lebih dari 1

### Perhitungan Skor

Perhitungan skor pada metode NWKNN dapat dihitung berdasarkan Persamaan 2.7.

**(2.7)**

Dimana:

: Bobot kelas

: Data latih pada kumpulan tetangga terdekat dari data uji

: Bernilai 1 jika jarak dan bernilai 0 jika nilai jarak ∉ .

: Jenis atau kelas

## Normalisasi Data

Proses normalisasi data menggunakan perhitungan berdasarkan Persamaan 2.8.

**(2.8)**

Keterangan:

: Nilai baru dari hasil normalisasi

: Nilai sebelum normalisasi

: Nilai minimum yang didapatkan melalui *dataset*

: Nilai maksimum yang didapatkan melalui *dataset*

## Pengukuran Performa Algoritma

Proses pengukuran algoritma pengujian akurasi melibatkan ketepatan system dalam menyimpulkan kelas dari suatu data yang dimana pengujian dilakukan dengan menggunakan Persamaan 2.9.

**(2.9)**

# METODOLOGI

Pada bagian ini menjelaskan langkah-langkah peneliti dalam melakukan penelitian mulai dari pengambilan data sampai perhitungan akurasi dari penerapan algoritma *Neighbor Weight K-Nearest Neighbor* (NWKNN), langkah-langkah sebagai berikut:

## Tipe Penelitian

Penelitian ini bersifat Non-implementatif Deskriptif merupakan   rancangan penelitian yang menitik beratkan pada investigasi terhadap situasi tertentu yang nantinya menghasilkan tinjauan ilmiah dengan melaksanakan tinjauan dan sintesis terhadap pengetahuan yang telah ada berupa kegiatan dalam menjelaskan karakteristik objek penelitian situasi tertentu yang sedang diteliti.

## Strategi Penelitian

Pada strategi penelitian untuk mengimplementasikan algoritma NWKNN untuk memberikan rekomendasi tema judul skripsi yang ingin diambil untuk mahasiswa Ilmu Keolahragaan Fakultas Ilmu Keolahragaan Universitas Negeri Medan. Data yang digunakan adalah data nilai mata kuliah wajib fakultas dan mata kuliah keminatan, yang diambil dari wisudawan angkatan 2016/2019 yang telah menyelesaikan studinya. Algoritma NKWNN ini dipilih karena konsep perhitungannya yang bersifat *lazy learner* yaitu dimana dicocokkan nilai mahasiswa yang ingin mencari tema dengan *dataset* yang tersedia. Dan dari hasil penelitian (Hadi, et al., 2018) yang membandingkan algoritma KNN dan NWKNN, dimana NWKNN menghasilkan nilai akurasi yang lebih baik 5% rata-rata dari setiap penelitian.

Tahapan pemrosesan data hingga menghasilkan hasil rekomendasi pada penelitian ini diawali dengan pengambilan data mahasiswa angkatan 2016/2019 yang telah tamat yang dijadikan sebagai *dataset* kemudian penggunaan *data testing* sebagai data yang ingin dicari pertimbangannya. Selanjutnya penentuan *cluster* dari tiap judul skripsi yang nantinya dijadikan tema yang diangkat dan menjadi kelas untuk proses klasifikasinya. Untuk teknik pengklasteran menggunakan algoritma K-Means dan program yang berasal dari (Kunwar, 2013) yang digunakan untuk pembagian tema-tema pada judul mahasiswa. Selanjutnya *dataset* yang telah dimasukkan dilakukan *preprocessing* dan normalisasi data. Dilanjutkan dengan penyeleksian mata kuliah yang digunakan untuk proses pengklasifikasian menggunakan teknik seleksi atribut *information gain*. Kemudian *data testing* yang ingin diberi rekomendasi dilakukan *preprocessing* dan dicari kecocokannya dengan *dataset,* pencarian kecocokan dengan menghitung nilai kedekatan tetangga menggunakan rumus *Euclidean Distance*. Kemudian dilakukan pengurutan perhitungan kedekatan tetangga. Setelah itu dikumpulkan kategori klasifikasi *nearest neighbor*. Kemudian dilakukan perhitungan bobot dari tiap kelas yang dilanjutkan dengan penghitungan skor untuk memberikan hasil akhir rekomendasi tema judul skripsi berdasarkan kelas dengan nilai skor tertinggi yang didapat. Selanjutnya dilakukan perhitungan akurasi dengan membandingkan nilai K dan E yang berubah ubah untuk mencapai hasil akurasi yang paling baik, sehingga mengetahui nilai K dan E yang paling baik untuk memperoleh akurasi maksimal.

***Dataset***

*Pre-processing Data*

Data *Traning*

*Clustering* Judul

Normalisasi Data

*Cluster*

Data *Testing*

*Euclidean Distance*

Nilai *d(x1, x2)*

Pengurutan jarak

Urutan Data

Penghitungan Bobot

Nilai

**Hasil**

**Akurasi**

Penghitungan Skor

Hasil Prediksi

Penghitungan

Akurasi

Gambar 3.1 Gambar Diagram Blok Prediksi Tema Judul Skripsi dengan Algoritma Neighbor Weight K-Nearest Neighbor (NWKNN)

## Lokasi Penelitian

Penelitian ini dilakukan pada Fakultas Ilmu Keolahragaan Universitas Negeri Medan yang beralamat di Jalan Willem Iskandar, Pasar V Medan Estate, Kenangan Baru, Kec. Percut Sei Tuan, Kabupaten Deli Serdang, Sumatera Utara 20221.

## Teknik Pengumpulan Data

Pengumpulan data penelitian berupa studi dokumen dari mahasiswa wisudawan 2016/2019 Fakultas Ilmu Keolahragaan Universitas Negeri Medan dengan mengajukan permintaan penggunaan data dokumentasi dari para alumni Fakultas kepada ketua jurusan untuk dijadikan data penelitian skripsi.

### Teknik Analisis Data

Analisis data adalah suatu proses maupun langkah yang dilakukan untuk mengetahui kondisi sebuah data yang nantinya menghasilkan informasi. Langkah-langkah berikut:

#### Data Cleaning

*Data Cleaning* dilakukan untuk mengisi missing value, mengidentifikasi *outlier*, menangani data noise, mengoreksi data yang tidak konsisten, dan menyelesaikan masalah redudansi data akibat integrasi data.

#### Data Transformation

Data transformasi digunakan untuk mengubah suatu data yang nantinya diperoleh data yang lebih berkualitas.

#### Data Reduction

Data Reduksi digunakan untuk mereduksi dimensi, atribut ataupun jumlah data yang digunakan.

### Penerapan Algoritma

Pada penerapan algoritma NWKNN dibuat menjadi sebuah sistem yang diuji tingkat hasil prediksinya dengan menghitung nilai akurasi sistem. Untuk mengetahui nilai akurasi tertinggi dilakukan percobaan terhadap nilai K dan E, untuk nilai K dan E yang memiliki akurasi tertinggi akan dijadikan sebagai nilai K dan E yang digunakan oleh sistem. Sehingga nantinya dapat menghasil hasil yang maksimal terdapat pemberian rekomendasi tema kepada mahasiswa.

## Peralatan Pendukung

Dalam penelitian ini menggunakan beberapa alat pendukung agar penelitian dapat berjalan dengan baik. Alat pendukung yang digunakan dalam bentuk perangkat keras (*Hardware*) dan perangkat lunak (*Software*).

### Perangkat Keras (*Hardware*)

Spesifikasi *hardware* yang digunakan dalam menjalankan penelitian ini sebagai berikut.

         *Processor Intel core* i-5

         Memori 8GB

### Perangkat Lunak (*Software*)

Spesifikasi *software* yang digunakan dalam menjalankan penelitian ini sebagai berikut.

* Sistem Operasi Windows 10 64-bit
* Netbeans IDE 8.1

# PERANCANGAN

Pada bagian ini menjelaskan mengenai perancangan algoritma, perancangan user interface dan langkah-langkah perhitungan manual dalam menyelesaikan masalah penelitian secara sistematik dalam pengimplementasian algoritma *Neighbor Weight K-Nearest Neighbor* (NWKNN) pada pemberian rekomendasi tema judul skripsi mahasiswa Jurusan Ilmu Keolahragaan Fakultas Ilmu Keolahragaan Universitas Negeri Medan.

## Diagram Alir Perancangan Algoritma

Pada bagian rancangan penelitian ini menggunakan penerapan dari algoritma *Neighbor Weight K-Nearest Neighbor* (NWKNN) dalam melakukan pemberian rekomendasi tema judul skripsi pada mahasiswa Ilmu Keolahragaan Universitas Negeri Medan. Dalam perancangan ini memiliki beberapa tahapan proses penyelesaian masalah pemberian rekomendasi yaitu bagian input *dataset,* persentasedata *training*, persentase data *testing*, nilai *K*, nilai *E*, bagian *preprocessing* data, bagian normalisasi data, pengklasteran data *training*, penyeleksian fitur yang digunakan, proses perhitungan *Euclidean*, pengurutan jarak, perhitungan bobot, perhitungan skor, dan perhitungan akurasi. Berikut diagram alir yang digunakan sebagai gambaran proses dari perancangan algoritma pada bagian Gambar 4.1.

Mulai

Input: *Dataset*,persentase data *training*, persentase data *testing*, banyak fitur, nilai *K*, nilai *E*

*Pre-processing* Data

*j* kurang dari (banyak\_fitur)

*Normalisasi* Data

Pengklasteran Data *Training*

Penyeleksian Fitur

Perhitungan jarak

A

A

Pengurutan jarak

Perhitungan Bobot

Output: Rekomendasi Tema, Hasil Akurasi

Perhitungan Akurasi

Perhitungan Skor

Selesai

Gambar 4.1 Diagram Alir Perancangan Algoritma

Dari diagram alir pada Gambar 4.1 dapat dijelaskan setiap langkah dari algoritma NWKNN sebagai berikut.

1. Memasukkan input dari sistem dalam bentuk user interface berupa nilai dataset berupa nilai mahasiswa, persentase dari data *training* dan *testing*, nilai *K* yang digunakan, nilai *E* yang digunakan.
2. Melakukan proses *pre-processing* data dengan metode analisis teknikal.
3. Melakukan proses normalisasi data dari yang sudah dimasukkan sebelumnya dengan rumus *Min-Max Normalization* sesuai dengan Persamaan 2.6.
4. Melakukan proses pengklasteran data *training* menggunakan algoritma *k-means* dengan menggunakan aplikasi dari (Kunwar, 2013).
5. Melakukan proses seleksi fitur menggunakan algoritma PSO pada Persamaan 2.1.
6. Melakukan proses perhitungan jarak menggunakan rumus *Euclidean Distance* sesuai dengan Persamaan 2.3.
7. Melakukan proses pengurutan nilai jarak menggunakan *library* algoritma *sorting*.
8. Melakukan perhitungan bobot nilai menggunakan rumus bobot pada Persamaan 2.4.
9. Melakukan proses perhitungan skor menggunakan rumus Skor dengan Persamaan 2.5.
10. Melakukan perhitungan akurasi dari sistem menggunakan rumus pada Persamaan 2.7.
11. Menghasilkan keluaran *user interface* berupa hasil rekomendasi tema judul skripsi dan akurasi sistem terhadap data *testing* yang digunakan.

### Diagram Alir *Pre-processing* Data

*Pre-processing* Data

Data[*i*][*j*] = dataset[*i*][*j*]

Menambahkan nilai *j* += 1

Mulai

Membaca *Dataset*

Membuat variable Data = *Double* [][]

Perulangan *i*=0 sampai kurang dari (panjang *dataset*)

Memberikan nilai *j*=0

A

*Input*: *Dataset*

j < Data[i]

*False*

*True*

B

B

A

*Output*: Data[*i*][*j*]

Selesai

Menambahkan nilai *i* += 1

Gambar 4.2 Diagram Alir Pre-processing Data

Dari diagram alir pada bagian Gambar 4.2 dapat dijelaskan setiap langkah dari *pre-processing* data sebagai berikut.

1. Memasukkan file *dataset* berupa *Excel* (.xls) dan banyak\_fitur yang digunakan.
2. Melakukan proses membaca dataset dengan menggunakan library jxl untuk membaca file dengan format *Extensible Stylesheet Language* (.xls).
3. Membuat *variable* Data dengan tipe *Double* *array* 2 dimensi.
4. Melakukan perulangan *variable* *i* dengan inisialisasi awal = 0 sampai kurang dari (panjang *dataset*).
5. Melakukan inisialisasi awal *variable* *j* bernilai 0.
6. Melakukan pengecekan nilai *j* yang kurang dari panjang *array* Data[*i*].
7. Melakukan penambahan nilai *j* dengan nilai 1.
8. Memasukkan nilai pada *array* Data sesuai dengan indeks baris = *i* dan indeks kolom = *j* dengan nilai dari dataset dengan baris = *i* dan kolom = *j*.
9. Menambahkan nilai *i* dengan nilai 1.
10. Menghasilkan keluaran berupa *array* Data 2 dimensi dengan tipe *String*.

### Diagram Alir Normalisasi Data

Normalisasi Data

Memberikan nilai *j*=0

Membuat variable *min* dan *max* = double

Mulai

Inisialisasi persentase data *training*, persentase data *testing*

Perulangan *i*=0 sampai kurang dari (panjang *array* Data)

j < Data[i]

*False*

*True*

*False*

*j* bernilai > 0

Memberikan nilai max dan min = Data[i][j]

*True*

*Data*[*i*][*j*]>max

*False*

*True*

Memberikan nilai *max* = *Data*[i][j]

*False*

*Data*[*i*][*j*]<min

*True*

Memberikan nilai min = *Data*[i][j]

A

D

C

B http://linkpoi.me/kFDKW

B

A

C

D

Menambahkan nilai *j* += 1

Menambahkan nilai *i* += 1

Membuat variable Normalisasi = double [][]

Perulangan *i*=0 sampai kurang dari (panjang Normalisasi)

Memberikan nilai *j*=0

*j* < normalisasi[i]

*True*

*False*

Normalisasi [*i*][j] = (Data[*i*][*j*]-min)/(max-min)

Menambahkan nilai *j* += 1

Menambahkan nilai *i* += 1

*Output*: Normalisasi[*i*][*j*]

Selesai

Gambar 4.3 Diagram Alir Normalisasi

Dari diagram alir pada Gambar 4.3 dapat dijelaskan setiap langkah dari proses normalisasi data sebagai berikut.

1. Memasukkan persentase data *training* dan data *testing* yang digunakan dari *array* Data [][] yang telah diambil.
2. Melakukan proses pembuatan *variable* min dan max dengan tipe data *double* untuk tempat penyimpanan nilai minimum dan maximum.
3. Melakukan perulangan *variable* *i* dengan inisialisasi awal = 0 sampai kurang dari (panjang *array* Data).
4. Melakukan inisialisasi awal *variable* *j* bernilai 0.
5. Melakukan pengecekan nilai *j* yang kurang dari panjang *array* Data[*i*].
6. Melakukan pengecekan jika nilai j lebih besar dari 0.
7. Jika pengecekan bernilai salah maka akan dibuat nilai min dan max bernilai sama dengan nilai dari Data[*i*][*j*] dan langsung menuju proses penambahan nilai *j*.
8. Melakukan pengecekan nilai maksimum.
9. Jika hasil pengecekan sebelumnya bernilai benar maka akan dilakukan pemberian nilai pada *variable* max sama dengan nilai dari *array* Data[*i*][*j*] dan jika pengecekan bernilai salah maka dilanjutkan ke proses pengecekan nilai minimal.
10. Melakukan pengecekan untuk mencari nilai minimal.
11. Jika bernilai benar maka akan dilakukan proses pemberian nilai *variable* min sama dengan nilai dari *array* Data[*i*][*j*] dan jika bernilai salah maka akan langsung menuju proses penambahan nilai *j*.
12. Melakukan penambahan nilai *j* dengan nilai 1.
13. Melakukan penambahan nilai *i* dengan nilai 1.
14. Membuat *variable* Normalisasi dengan tipe *Double* *array* 2 dimensi.
15. Melakukan perulangan *variable* *i* dengan inisialisasi awal = 0 sampai kurang dari (panjang *dataset*).
16. Melakukan inisialisasi awal *variable* *j* bernilai 0.
17. Melakukan pengecekan nilai *j* yang kurang dari panjang *array* Data[*i*].
18. Memberikan nilai dari Normalisasi[*i*][*j*] = (Data[*i*][*j*]-min)/(max-min).
19. Melakukan penambahan nilai *j* dengan nilai 1.
20. Melakukan penambahan nilai *i* dengan nilai 1.
21. Memasukkan nilai pada *array* Normalisasi sesuai dengan indeks baris = i dan indeks kolom = j dengan nilai dari dataset dengan baris = i dan kolom = j.
22. Menghasilkan keluaran berupa *array* Normalisasi 2 dimensi dengan tipe data *Double*.

### Diagram Alir Penyeleksian Fitur

Penyeleksian Fitur

Mulai

Inisialisasi Nilai K, E, banyak iterasi, , , , , , *swarm* dan banyak partikel

Inisialisasi Populasi dan Kecepatan

*Looping* banyak generasi (iterasi)

Evaluasi *Fitness*

*i* < swarm

*False*

i += 1

*True*

<

*False*

*True*

=

Penyimpanan nilai terbaik

**B**

**A**

A

B

Update kecepatan dengan Persamaan 2.1 dari perulangan sebelumnya

Update Posisi Partikel

Iterasi < generasi

*False*

*True*

Stop

Gambar 4.4 Diagram Alir Seleksi Fitur

Dari diagram alir pada Gambar 4.4 dapat dijelaskan setiap langkah dari proses penyeleksian fitur sebagai berikut.

1. Inisialisasi Nilai K, E, banyak iterasi (generasi) yang digunakan, , , , , , *swarm* dan banyak partikel yang digunakan.
2. Melakukan pembuatan populasi menggunakan rand [0, 1], dan membuat kecepatan minimal dan maximal partikel dengan nilai [-0.6, 0.6].
3. Membuat perulangan berdasarkan banyak generasi.
4. Melakukan pengecekan banyak partikel yang digunakan.
5. Melakukan pengecekan nilai fitness tertinggi dari sebelumnya dengan partikel sekarang.
6. Melakukan pengupdatan nilai tertinggi setiap partikel dari kumpulan partikel.
7. Melakukan perubahan kecepatan dengan menggunakan Persamaan 2.1 yang didapat dari perulangan sebelumnya.
8. Melakukan perubahan posisi partikel.
9. Melakukan pengecekan apakah sudah mencapai generasi yang diinginkan.

### Diagram Alir Perhitungan Jarak

Perhitungan Jarak

Mulai

Membuat variabel *jarak* = *double* []

Menambahkan nilai *j* += 1

Normalisasi Data

*Input*: DataUji

Perulangan *i*=0 sampai kurang dari panjang normaliasi

Memberikan nilai *j* dan *jarak* =0

*False*

*j* < normalisasi[i]

*True*

Menambahkan nilai jarak +=

Menambahkan nilai *i* += 1

Menambahkan nilai jarak [i] = *jarak*

*Output*: *Jarak* []

Stop

Gambar 4.5 Diagram Alir Perhitungan Jarak

Dari diagram alir pada Gambar 4.5 dapat dijelaskan setiap langkah dari proses perhitungan jarak sebagai berikut.

1. Memasukkan data uji yang ingin dicari jaraknya.
2. Melakukan proses normalisasi.
3. Membuat *variable* jarak dengan tipe *Double* *array* 1 dimensi.
4. Melakukan perulangan *variable* *i* dengan inisialisasi awal = 0 sampai kurang dari (panjang *dataset*).
5. Melakukan inisialisasi awal *variable* *j* bernilai 0.
6. Melakukan pengecekan nilai *j* yang kurang dari panjang *array* normalisasi[*i*].
7. Memberikan nilai jarak dengan penambahan jarak dari setiap fitur.
8. Melakukan penambahan nilai *j* dengan nilai 1.
9. Memasukkan nilai jarak pada *array* jarak sesuai dengan indeks baris = i.
10. Melakukan penambahan nilai *i* dengan nilai 1.
11. Menghasilkan keluaran *array* jarak.

### Diagram Alir Perhitungan Bobot

Perhitungan Bobot

Mulai

Mengambil variable sort\_jarak, *K*, *E*

Mengambil nilai K teratas dari sort\_jarak

Membuat variabel bobot = *double* [] dan hitung = int []

Perulangan *i*=0 sampai kurang dari banyak kelas

**B**

**A**

**B**

**A**

Membuat nilai *j* = 0

*False*

*j* < *K*

*True*

Label-1 == *i*

hitung [i] += 1

Membuat nilai *j* += 1

Menambahkan Nilai bobot *i* = (1 / ((hitung[i]) / min(hitung))^2)

Menambahkan nilai *i* += 1

*Output*: bobot []

Stop

Gambar 4.6 Diagram Alir Pembobotan

Dari diagram alir pada Gambar 4.6 dapat dijelaskan setiap langkah dari proses perhitungan jarak sebagai berikut.

1. Mengambil data dari *variable* jarak yang telah di sort, dan mengambil nilai *K* dan *E* untuk perhitungan.
2. Mengambil data berdasarkan nilai k dari atas tabel.
3. Melakukan pembuatan *variable* bobot untuk menyimpan hasil bobot dan *variable* hitung untuk menyimpan banyak data label tertentu dalam urutan *K* tersebut.
4. Membuat perulangan untuk mencari nilai dari bobot setiap label.
5. Membuat *variable* j dengan nilai 0
6. Melakukan pengecekan nilai yang lebih kecil dari *variable* *j* yang dibandingkan dengan nilai *K*.
7. Melakukan pengecekan label pada data dengan mengurangi 1 dan menyamakannya dengan indeks dari *variable* *i*.
8. Melakukan penambahan *variable* hitung dengan nilai 1.
9. Melakukan penambahan *variable* *j* dengan nilai 1.
10. Melakukan Penambahan nilai pada *variable* bobot dengan Persamaan 2.2.
11. Melakukan penambahan *variable* *i* dengan nilai 1.
12. Menghasilkan *variable* bobot yang telah diisi.

### Diagram Alir Perhitungan Skor

Perhitungan Skor

Mulai

Mengambil variable bobot [], nilai *sort*\_jarak [] sesuai K, hitung [],

Membuat variabel skor = *double* [], hitungS = *double* [], C = double

Perulangan *i*=0 sampai kurang dari banyak kelas

**B**

**A**

**B**

**A**

Membuat nilai *j* = 0

*False*

*j* < *K*

Jarak(label)-1 == *i*

*True*

*False*

*True*

hitungS[i] += jarak(label)

Membuat nilai *j* += 1

hitung[i]>0

*False*

C= 0

*True*

C= 1

hitungS[i] += 1

Menambahkan Nilai skor [*i*] = hitungS[i]\*(bobot[*i*]\*C)

Menambahkan nilai *i* += 1

**B**

**A**

**B**

**A**

*Output*: skor [] dan Kelas

Stop

Gambar 4.7 Diagram Alir Penentuan Skor

Dari diagram alir pada Gambar 4.7 dapat dijelaskan setiap langkah dari proses perhitungan jarak sebagai berikut.

1. Mengambil data dari *variable* jarak yang telah di sort dengan nilai *K*, nilai dari *variable array* hitung, nilai dari *variable* bobot.
2. Melakukan pembuatan *variable* skor untuk menyimpan hasil skor dan *variable* hitungS untuk menyimpan total jarak dari label tertentu dan nilai C untuk melihat adakah anggota kelas dalam urutan *K* tersebut Membuat perulangan untuk mencari nilai dari bobot setiap label.
3. Membuat perulangan untuk mencari nilai dari skor setiap label.
4. Membuat *variable* j dengan nilai 0
5. Melakukan pengecekan nilai yang lebih kecil dari *variable* *j* yang dibandingkan dengan nilai *K*.
6. Melakukan pengecekan label pada data dengan mengurangi 1 dan menyamakannya dengan indeks dari *variable* *i*.
7. Melakukan penambahan *variable* hitungS dengan nilai jarak label.
8. Melakukan penambahan *variable* *j* dengan nilai 1.
9. Melakukan pengecekan keanggotaan label untuk melihat label yang tidak memiliki anggota
10. Memberikan penilaian pada nilai C dengan nilai 1 untuk label yang memiliki anggota
11. Memberikan penilaian pada nilai C dengan nilai 0 untuk label yang tidak memiliki anggota.
12. Melakukan Penambahan nilai pada *variable* skor dengan Persamaan 2.3.
13. Melakukan penambahan *variable* *i* dengan nilai 1.
14. Menghasilkan *variable* skor yang telah diisi dan kelas dari data uji.

## Perhitungan Manualisasi

Pada proses di bagian perhitungan manualisasi dilakukan tahap untuk menghitung setiap langkah-langkah yang dikerjakan oleh algoritma secara manual. Pada perhitungan ini menjelaskan langkah-langkah manual dari algoritma NWKNN untuk kasus pemberian rekomendasi. Dalam melakukan perhitungan proses klasifikasi untuk pemberian rekomendasi tema untuk judul skripsi mahasiswa Fakultas Ilmu Keolahragaan Fakultas Ilmu Keolahragaan Universitas Negeri Medan. Sebelum menuju proses algoritma NWKNN diperlukan persiapan data sebagai contoh awal dengan diambil 5 Mata kuliah dengan 30 data sebagai contoh perhitungan manualisasi dapat dilihat pada Tabel 4.1.

Tabel 4.1 Data Nilai Mata kuliah Mahasiswa dan Judul Skripsi

Tabel 4.2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | MK1 | MK2 | MK3 | MK4 | MK5 |
| 1 | C | C | C | C | B |
| 2 | C | B | C | C | A |
| 3 | B | C | C | C | C |
| 4 | B | B | C | C | C |
| 5 | C | B | A | B | B |
| 6 | C | C | A | C | C |
| 7 | B | B | C | C | C |
| 8 | B | B | C | A | B |
| 9 | B | C | B | C | C |
| 10 | B | B | C | C | C |
| 11 | B | B | C | C | C |
| 12 | C | B | B | C | A |
| 13 | C | B | B | C | C |
| 14 | B | B | B | B | C |
| 15 | B | B | A | B | C |
| 16 | B | B | B | C | C |
| 17 | B | B | B | A | C |
| 18 | C | B | B | C | C |
| 19 | B | A | B | B | C |
| 20 | B | C | C | B | C |
| 21 | B | B | A | C | C |
| 22 | C | B | C | A | B |
| 23 | B | C | C | A | C |
| 24 | A | B | B | B | B |
| 25 | B | B | A | A | A |
| 26 | C | B | A | C | B |
| 27 | B | B | C | C | C |
| 28 | B | B | B | B | B |
| 29 | C | C | C | C | B |
| 30 | C | B | B | C | C |

### Inisialisasi Data Numerik, Data *Training* dan Data *Testing*

Sebelum data diatas diproses dilakukan perubahan bentuk nilai mahasiswa menjadi bentuk nilai numerik yang berdasarkan ketentuan Fakultas Ilmu Keolahragaan yang mengacu pada Tabel 4.2.

Tabel 4.3 Konversi Nilai Huruf

|  |  |
| --- | --- |
| Huruf | Angka |
| A | 4 |
| B | 3 |
| C | 2 |
| D | 1 |

Setelah dilakukan perubahan maka menghasilkan data nilai mahasiswa berupa nilai numerik yang ditunjukan pada Tabel 4.3.

Tabel 4.4 Hasil Konversi Nilai Mata kuliah Mahasiswa

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | MK1 | MK2 | MK3 | MK4 | MK5 |
| 1 | 2 | 2 | 2 | 2 | 3 |
| 2 | 2 | 3 | 2 | 2 | 4 |
| 3 | 3 | 2 | 2 | 2 | 2 |
| 4 | 3 | 3 | 2 | 2 | 2 |
| 5 | 2 | 3 | 4 | 3 | 3 |
| 6 | 2 | 2 | 4 | 2 | 2 |
| 7 | 3 | 3 | 2 | 2 | 2 |
| 8 | 3 | 3 | 2 | 4 | 3 |
| 9 | 3 | 2 | 3 | 2 | 2 |
| 10 | 3 | 3 | 2 | 2 | 2 |
| 11 | 3 | 3 | 2 | 2 | 2 |
| 12 | 2 | 3 | 3 | 2 | 4 |
| 13 | 2 | 3 | 3 | 2 | 2 |
| 14 | 3 | 3 | 3 | 3 | 2 |
| 15 | 3 | 3 | 4 | 3 | 2 |
| 16 | 3 | 3 | 3 | 2 | 2 |
| 17 | 3 | 3 | 3 | 4 | 2 |
| 18 | 2 | 3 | 3 | 2 | 2 |
| 19 | 3 | 4 | 3 | 3 | 2 |
| 20 | 3 | 2 | 2 | 3 | 2 |
| 21 | 3 | 3 | 4 | 2 | 2 |
| 22 | 2 | 3 | 2 | 4 | 3 |
| 23 | 3 | 2 | 2 | 4 | 2 |
| 24 | 4 | 3 | 3 | 3 | 3 |
| 25 | 3 | 3 | 4 | 4 | 4 |
| 26 | 2 | 3 | 4 | 2 | 3 |
| 27 | 3 | 3 | 2 | 2 | 2 |
| 28 | 3 | 3 | 3 | 3 | 3 |
| 29 | 2 | 2 | 2 | 2 | 3 |
| 30 | 2 | 3 | 3 | 2 | 2 |

Setelah itu dilakukan pembentukan data *training* dan data *testing* dengan perbandingan 70%:30% dari data yang digunakan, sehingga menghasilkan data *training* dan data *testing* yang terdapat pada Tabel 4.4 dan Tabel 4.5.

Tabel 4.5 Data *Training*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 2 | 2 | 2 | 3 |
| 2 | 2 | 3 | 2 | 2 | 4 |
| 3 | 3 | 2 | 2 | 2 | 2 |
| 4 | 3 | 3 | 2 | 2 | 2 |
| 5 | 2 | 3 | 4 | 3 | 3 |
| 6 | 2 | 2 | 4 | 2 | 2 |
| 7 | 3 | 3 | 2 | 2 | 2 |
| 8 | 3 | 3 | 2 | 4 | 3 |
| 9 | 3 | 2 | 3 | 2 | 2 |
| 10 | 3 | 3 | 2 | 2 | 2 |
| 11 | 3 | 3 | 2 | 2 | 2 |
| 12 | 2 | 3 | 3 | 2 | 4 |
| 13 | 2 | 3 | 3 | 2 | 2 |
| 14 | 3 | 3 | 3 | 3 | 2 |
| 15 | 3 | 3 | 4 | 3 | 2 |
| 16 | 3 | 3 | 3 | 2 | 2 |
| 17 | 3 | 3 | 3 | 4 | 2 |
| 18 | 2 | 3 | 3 | 2 | 2 |
| 19 | 3 | 4 | 3 | 3 | 2 |
| 20 | 3 | 2 | 2 | 3 | 2 |
| 21 | 3 | 3 | 4 | 2 | 2 |

Tabel 4.6 Data *Testing*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | MK1 | MK2 | MK3 | MK4 | MK5 |
| 1 | 2 | 2 | 2 | 2 | 3 |
| 2 | 2 | 3 | 2 | 2 | 4 |
| 3 | 3 | 2 | 2 | 2 | 2 |
| 4 | 3 | 3 | 2 | 2 | 2 |
| 5 | 2 | 3 | 4 | 3 | 3 |
| 6 | 2 | 2 | 4 | 2 | 2 |
| 7 | 3 | 3 | 2 | 2 | 2 |
| 8 | 3 | 3 | 2 | 4 | 3 |
| 9 | 3 | 2 | 3 | 2 | 2 |

### Perhitungan Normalisasi

Sebelum data dilakukan proses perhitungan dilakukan terlebih dahulu proses menyeimbangkan data dengan normalisasi. Dalam normalisasi data ini digunakan perhitungan *min-max normalization*. Pada *min-max normalization* terlebih dahulu harus menentukan nilai minimum dan nilai maksimum, nilai minimum didapat dengan mencari nilai terendah dari keseluruhan data *training* dan data *testing* dan untuk nilai maksimum didapat dengan mencari nilai tertinggi dari keseluruhan data *training* dan data *testing* dan jika data yang ada lebih rendah dan lebih tinggi dari nilai min dan max maka akan dilakukan *update* nilai min dan max secara otomatis. Sehingga dari data diatas didapat nilai minimum dan maksimum yang dapat dilihat pada Tabel 4.6.

Tabel 4.7 Nilai Minimum dan Maksimum

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Minimum | 2 |
| Maksimum | 4 |

Proses perhitungan selanjutnya adalah menghitung normalisasi data menggunakan sesuai rumus pada Persamaan 2.6. Berikut hasil perhitungan manual untuk normalisasi data training pada data ke-1 dan mata kuliah M1.

Hasil detail normalisasi dapat diamati pada bagian Tabel 4.7 dan Tabel 4.6.

Tabel 4.8 Normalisasi Data *Training*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | MW1 | MW2 | MW3 | MW4 | MW5 |
| 1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.5 |
| 2 | 0.0 | 0.5 | 0.0 | 0.0 | 1.0 |
| 3 | 0.5 | 0.0 | 0.0 | 0.0 | 0.0 |
| 4 | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 |
| 5 | 0.0 | 0.5 | 1.0 | 0.5 | 0.5 |
| 6 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| 7 | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 |
| 8 | 0.5 | 0.5 | 0.0 | 1.0 | 0.5 |
| 9 | 0.5 | 0.0 | 0.5 | 0.0 | 0.0 |
| 10 | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 |
| 11 | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 |
| 12 | 0.0 | 0.5 | 0.5 | 0.0 | 1.0 |
| 13 | 0.0 | 0.5 | 0.5 | 0.0 | 0.0 |
| 14 | 0.5 | 0.5 | 0.5 | 0.5 | 0.0 |
| 15 | 0.5 | 0.5 | 1.0 | 0.5 | 0.0 |
| 16 | 0.5 | 0.5 | 0.5 | 0.0 | 0.0 |
| 17 | 0.5 | 0.5 | 0.5 | 1.0 | 0.0 |
| 18 | 0.0 | 0.5 | 0.5 | 0.0 | 0.0 |
| 19 | 0.5 | 1.0 | 0.5 | 0.5 | 0.0 |
| 20 | 0.5 | 0.0 | 0.0 | 0.5 | 0.0 |
| 21 | 0.5 | 0.5 | 1.0 | 0.0 | 0.0 |

Tabel 4.9 Normalisasi Data *Testing*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | MW1 | MW2 | MW3 | MW4 | MW5 |
| 1 | 0.0 | 0.5 | 0.0 | 1.0 | 0.5 |
| 2 | 0.5 | 0.0 | 0.0 | 1.0 | 0.0 |
| 3 | 1.0 | 0.5 | 0.5 | 0.5 | 0.5 |
| 4 | 0.5 | 0.5 | 1.0 | 1.0 | 1.0 |
| 5 | 0.0 | 0.5 | 1.0 | 0.0 | 0.5 |
| 6 | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 |
| 7 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| 8 | 0.0 | 0.0 | 0.0 | 0.0 | 0.5 |
| 9 | 0.0 | 0.5 | 0.5 | 0.0 | 0.0 |

### Pengklasteran Data

Sebelum melakukan klasifikasi data dilakukan pengklusteran untuk membuat kelompok-kelompok dari setiap judul-judul skripsi mahasiswa sehingga menjadi tema yang direkomendasikan kepada mahasiswa. Pengklasteran pada penelitian ini menggunakan library *K-Means* dengan nilai banyak kluster yang dibuat sebanyak 3, sehingga menghasilkan data *training* yang telah diberi pelabelan cluster dapat dilihat pada Tabel 4.9.

Tabel 4.10 Pemberian Label Pada Data Latih

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | 1 | 2 | 3 | 4 | 5 | Cluster |
| 1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.5 | 2 |
| 2 | 0.0 | 0.5 | 0.0 | 0.0 | 1.0 | 2 |
| 3 | 0.5 | 0.0 | 0.0 | 0.0 | 0.0 | 2 |
| 4 | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 | 1 |
| 5 | 0.0 | 0.5 | 1.0 | 0.5 | 0.5 | 3 |
| 6 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 3 |
| 7 | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 | 2 |
| 8 | 0.5 | 0.5 | 0.0 | 1.0 | 0.5 | 2 |
| 9 | 0.5 | 0.0 | 0.5 | 0.0 | 0.0 | 1 |
| 10 | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 | 2 |
| 11 | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 | 3 |
| 12 | 0.0 | 0.5 | 0.5 | 0.0 | 1.0 | 3 |
| 13 | 0.0 | 0.5 | 0.5 | 0.0 | 0.0 | 2 |
| 14 | 0.5 | 0.5 | 0.5 | 0.5 | 0.0 | 1 |
| 15 | 0.5 | 0.5 | 1.0 | 0.5 | 0.0 | 1 |
| 16 | 0.5 | 0.5 | 0.5 | 0.0 | 0.0 | 3 |
| 17 | 0.5 | 0.5 | 0.5 | 1.0 | 0.0 | 2 |
| 18 | 0.0 | 0.5 | 0.5 | 0.0 | 0.0 | 1 |
| 19 | 0.5 | 1.0 | 0.5 | 0.5 | 0.0 | 3 |
| 20 | 0.5 | 0.0 | 0.0 | 0.5 | 0.0 | 3 |
| 21 | 0.5 | 0.5 | 1.0 | 0.0 | 0.0 | 1 |
| 22 | 0.0 | 0.5 | 0.0 | 1.0 | 0.5 | 1 |
| 23 | 0.5 | 0.0 | 0.0 | 1.0 | 0.0 | 2 |
| 24 | 1.0 | 0.5 | 0.5 | 0.5 | 0.5 | 2 |
| 25 | 0.5 | 0.5 | 1.0 | 1.0 | 1.0 | 3 |
| 26 | 0.0 | 0.5 | 1.0 | 0.0 | 0.5 | 1 |
| 27 | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 | 1 |
| 28 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 2 |
| 29 | 0.0 | 0.0 | 0.0 | 0.0 | 0.5 | 3 |
| 30 | 0.0 | 0.5 | 0.5 | 0.0 | 0.0 | 3 |

### Penyeleksian Fitur

Sebelum melakukan klasifikasi dilakukan penyeleksian fitur yang digunakan, fitur pada kasus ini merupakan mata kuliah yang digunakan sebagai bahan perhitungan untuk melihat jarak terdekat dari data *testing*. Untuk penyeleksian fitur digunakan algoritma *Binary Particle Swarm Optimization* (BPSO), dalam melakukan penyeleksian fitur menggunakan algoritma BPSO dilakukan inisialisasi awal banyak partikel dan panjang partikel. Untuk partikel yang digunakan sebagai contoh sebanyak 2 dan untuk panjang partikel sebanyak 5 fitur.

#### Inisialisasi Awal Partikel

Dalam melakukan seleksi dibuat sebuah partikel yang merepresentasikan diambilnya nilai mata kuliah dengan nilai 1 dan nilai 0 tidak diambilnya nilai mata kuliah. Sehingga menghasilkan partikel dengan panjang 5 dengan nilai r1 dan r2 merupakan bilangan acak dalam interval [0, 1]. Untuk *fitness* yang digunakan merupakan hasil akurasi dari pengukuran dengan mencocokan hasil klasifikasi NWKNN dengan *K* = 5 dan *E* = 2. Sehingga menghasilkan inisialisasi partikel awal yang terdapat pada Tabel 4.10.

Tabel 4.11 Inisialisasi Awal Partikel

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Partikel | Matakuliah | | | | | Fitness |
| 1 | 2 | 3 | 4 | 5 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0.00 |
| 2 | 1 | 0 | 0 | 1 | 0 | 0.00 |
| 3 | 1 | 0 | 0 | 1 | 1 | 0.33 |

#### Inisialisasi Pbest dan Gbest

Pada iterasi awal atau perulangan ke-0 (*t*=0), maka nilai Pbest disamakan dengan nilai posisi awal partikel, yaitu ( dan konversi partikel menjadi seperti Tabel 4.11.

Tabel 4.12 Pbest Awal

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Matakuliah | | | | | Fitness |
| 1 | 2 | 3 | 4 | 5 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0.00 |
| 2 | 1 | 0 | 0 | 1 | 0 | 0.00 |
| 3 | 1 | 0 | 0 | 1 | 1 | 0.33 |

, dicari dengan memilih satu yang *fitness*-nya tertinggi sehingga nilai

, yang terdapat pada Tabel 4.12.

Tabel 4.13 Gbest Awal

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Matakuliah | | | | | Fitness |
| 1 | 2 | 3 | 4 | 5 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0.33 |

#### Update Kecepatan

*Update* Kecepatan dilakukan untuk menentukan arah perpindahan posisi partikel yang ada di populasi. Batasan *lower* dan *upper* kecepatan yang digunakan dalam proses ini berdasarkan nilai maksimum dari posisi partikel (*xmax*). Diketahui nilai *lower* dan *upper* untuk = [*xmin*, *xmax*] = [0, 1] karena memang nilai tersebut biner, dan untuk untuk = [*vm in*, *vmax*], dimana *vmin*=-*vmax* dan *vmax*=60%\**xmax.* Dalam menentukan interval ini, baik untuk posisi maupun kecepatan partikel, sebaiknya dilakukan pada proses inisialisasi (Cholissodin & Riyandani, 2016).

Sehingga menghasilkan nilai dan misalkan diketahui nilai,,, , dan . Dimana dan adalah bilangan acak interval. Maka untuk mendapatkan hasil *update* kecepatan dihitung dengan menggunakan Persamaan 2.1 sehingga menghasilkan nilai berikut.

Hasil dari *update* kecepatan yang diperoleh dapat dilihat dari Tabel 4.13.

Tabel 4.14 Hasil *Update* Kecepatan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Partikel | Matakuliah | | | | |
| 1 | 2 | 3 | 4 | 5 |
| 1 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| 2 | -0.40 | 0.40 | 0.40 | 0.00 | 0.00 |
| 3 | -0.40 | 0.40 | 0.40 | 0.00 | -0.40 |

#### Update Posisi

Setelah melakukan perhitungan kecepatan, kemudian dilakukan perhitungan sigmoid dengan Persamaan 2.2 sehingga menghasilkan perhitungan dan tabel sigmoid seperti Tabel 4.14.

Tabel 4.15 Sigmoid

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Partikel | Matakuliah | | | | |
| 1 | 2 | 3 | 4 | 5 |
| 1 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| 2 | 0.4 | 0.6 | 0.6 | 0.5 | 0.5 |
| 3 | 0.4 | 0.6 | 0.6 | 0.5 | 0.4 |

Selanjutnya membuat nilai yang terdapat pada Tabel 4.15, lalu menghitung hasil *update* posisi dan fitness menggunakan Persamaan 2.3 dan menghasilkan posisi baru pada Tabel 4.16.

Tabel 4.16 Nilai Random

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Partikel | Nilai Random | | | | |
| 1 | 2 | 3 | 4 | 5 |
| 1 | 0.8 | 0.6 | 0.7 | 0.2 | 0.6 |
| 2 | 0.1 | 0.4 | 0.8 | 0.9 | 0.5 |
| 3 | 0.7 | 0.8 | 0.3 | 0.8 | 0.5 |

Tabel 4.17 *Update* Posisi (t=1)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Partikel | Matakuliah | | | | | Fitness |
| 1 | 2 | 3 | 4 | 5 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0.00 |
| 2 | 1 | 1 | 0 | 0 | 1 | 0.00 |
| 3 | 0 | 0 | 1 | 0 | 0 | 0.33 |

#### Update Pbest dan Gbest

Selanjutnya dilakukan perbandingan menggunakan Persamaan 2.4 yang diambil nilai fitness tertingga apabila nilai fitnessnya yang sama maka diambil sebelumnya begitu juga dengan nilai dibandingkan nilai sebelumnya dengan nilai tertinggi jika nilai fitnessnya sama maka diambil nilai sebelumnya. Hasil *update* dan dapat dilihat pada Tabel 4.17 dan Tabel 4.18.

Tabel 4.18 *Update* *Pbest*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Partikel | Matakuliah | | | | | Fitness |
| 1 | 2 | 3 | 4 | 5 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0.0000 |
| 2 | 1 | 0 | 0 | 1 | 0 | 0.0000 |
| 3 | 1 | 0 | 0 | 1 | 1 | 0.3333 |

Tabel 4.19 Update *Gbest*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Partikel | Matakuliah | | | | | Fitness |
| 1 | 2 | 3 | 4 | 5 |
| 1 | 1 | 0 | 0 | 1 | 1 | 0.33 |

#### Kondisi Berhenti dan Fitur Digunakan

Kondisi berhenti melakukan pencarian nilai fitness terbaik sebanyak 10 perulangan sehingga menghasilkan nilai fitness tertinggi dan partikel yang dapat dilihat pada Tabel 4.19.

Tabel 4.20 Hasil Terbaik

|  |  |
| --- | --- |
| Partikel | Fitness |
| [1 0 0 1 1] | 0.33 |

Dari hasil Tabel 4.16 maka yang dipakai untuk menjadi fitur adalah mata kuliah Ke-1, 2, 4 dan ke-5 karena bernilai 1.

### Perhitungan Jarak

Untuk tahap awal klasifikasi dimulai dengan menghitung jarak antara data uji dengan data latih menggunakan Persamaan 2.5 dengan menggunakan data uji 1 sehingga menghasilkan perhitungan seperti dibawah.

1.118

Hasil dari perhitungan jarak data uji dengan data latih 1 dapat dilihat pada Tabel 4.20.

Tabel 4.21 Hasil Jarak Data Uji 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | Total Jarak | Kelas |
| 0.000 | 0.250 | 0.000 | 1.000 | 0.000 | 1.118 | 2 |
| 0.000 | 0.000 | 0.000 | 1.000 | 0.250 | 1.118 | 2 |
| 0.250 | 0.250 | 0.000 | 1.000 | 0.250 | 1.323 | 2 |
| 0.250 | 0.000 | 0.000 | 1.000 | 0.250 | 1.225 | 1 |
| 0.000 | 0.000 | 1.000 | 0.250 | 0.000 | 1.118 | 3 |
| 0.000 | 0.250 | 1.000 | 1.000 | 0.250 | 1.581 | 3 |
| 0.250 | 0.000 | 0.000 | 1.000 | 0.250 | 1.225 | 2 |
| 0.250 | 0.000 | 0.000 | 0.000 | 0.000 | 0.500 | 2 |
| 0.250 | 0.250 | 0.250 | 1.000 | 0.250 | 1.414 | 1 |
| 0.250 | 0.000 | 0.000 | 1.000 | 0.250 | 1.225 | 2 |
| 0.250 | 0.000 | 0.000 | 1.000 | 0.250 | 1.225 | 3 |
| 0.000 | 0.000 | 0.250 | 1.000 | 0.250 | 1.225 | 3 |
| 0.000 | 0.000 | 0.250 | 1.000 | 0.250 | 1.225 | 2 |
| 0.250 | 0.000 | 0.250 | 0.250 | 0.250 | 1.000 | 1 |
| 0.250 | 0.000 | 1.000 | 0.250 | 0.250 | 1.323 | 1 |
| 0.250 | 0.000 | 0.250 | 1.000 | 0.250 | 1.323 | 3 |
| 0.250 | 0.000 | 0.250 | 0.000 | 0.250 | 0.866 | 2 |
| 0.000 | 0.000 | 0.250 | 1.000 | 0.250 | 1.225 | 1 |
| 0.250 | 0.250 | 0.250 | 0.250 | 0.250 | 1.118 | 3 |
| 0.250 | 0.250 | 0.000 | 0.250 | 0.250 | 1.000 | 3 |
| 0.250 | 0.000 | 1.000 | 1.000 | 0.250 | 1.581 | 1 |

### Pengurutan Jarak

Setelah dilakukan pencarian jarak kemudian dilakukan pengurutan jarak dari yang terendah menuju yang tertinggi karena yang diambil adalah data yang memiliki kedekatan dengan data uji dengan menggunakan library sort yang tersedia sehingga menghasilkan urutan nilai seperti Tabel 4.21.

Tabel 4.22 Pengurutan Jarak

|  |  |  |
| --- | --- | --- |
| No | Total Jarak | Kelas |
| 1 | 0.500 | 2 |
| 2 | 0.866 | 2 |
| 3 | 1.000 | 1 |
| 4 | 1.000 | 3 |
| 5 | 1.118 | 2 |
| 6 | 1.118 | 2 |
| 7 | 1.118 | 3 |
| 8 | 1.118 | 3 |
| 9 | 1.225 | 1 |
| 10 | 1.225 | 2 |
| 11 | 1.225 | 2 |
| 12 | 1.225 | 3 |
| 13 | 1.225 | 3 |
| 14 | 1.225 | 2 |
| 15 | 1.225 | 1 |
| 16 | 1.323 | 2 |
| 17 | 1.323 | 1 |
| 18 | 1.323 | 3 |
| 19 | 1.414 | 1 |
| 20 | 1.581 | 3 |
| 21 | 1.581 | 1 |

### Perhitungan Bobot

Pada proses ini dilakukan perhitungan bobot untuk setiap label yang digunakan menggunakan Persamaan 2.6 dimana nilai min(jumlah anggota) merupakan banyak anggota dari kelas yang memiliki anggota paling sedikit yang tidak bernilai 0,dengan nilai *K*=5 dan *E*=2 yang diambil dari Tabel 4.18 sehingga menghasilkan perhitungan bobot untuk label 1 sebagai berikut berikut.

Setelah menghitung nilai dengan rumus sebelumnya kepada kelas yang lain sehingga menghasilkan tabel yang dapat dilihat pada Tabel 4.22.

Tabel 4.23 *Weight* Kelas

|  |  |
| --- | --- |
| Kelas | Bobot |
| 1 | 1 |
| 2 | 0.5 |
| 3 | 1 |

### Perhitungan Skor

Selanjutnya dilakukan perhitungan skor untuk melihat nilai skor tertinggi yang dijadikan sebagai label/kelas untuk data uji tersebut dengan menggunakan Persamaan 2.7 sehingga menghasilkan perhitungan skor untuk data uji 1 dan label 1 sebagai berikut.

Setelah menghitung nilai dengan rumus sebelumnya kepada kelas yang lain sehingga menghasilkan tabel yang dapat dilihat pada Tabel 4.23.

Tabel 4.24 Skor Kelas

|  |  |
| --- | --- |
| Kelas | Skor |
| 1 | 1 |
| 2 | 1.801047 |
| 3 | 1 |

Dari hasil Tabel 4.20 didapat nilai Skor kelas Ke-2 memiliki nilai tertinggi sehingga kelas untuk data latih 1 adalah kelas Ke-2.

### Penghitungan Performa

Untuk melakukan tingkat performa atau akurasi dari sistem dilakukan perhitungan menggunakan Persamaan 2.9 dan dengan nilai *K*=5, *E*=2 dan seleksi fitur yang telah dicari sehingga menghasilkan Tabel perbandingan kelas 4.24.

Tabel 4.25 Perbandingan Hasil Kelas

|  |  |  |
| --- | --- | --- |
| Data Ke- | Kelas Hasil Sistem | Kelas Asli |
| 1 | 2 | 1 |
| 2 | 3 | 2 |
| 3 | 2 | 3 |
| 4 | 2 | 2 |
| 5 | 2 | 2 |
| 6 | 1 | 1 |
| 7 | 3 | 2 |
| 8 | 3 | 1 |
| 9 | 2 | 1 |

Berdasarkan Tabel 4.21 sehingga menghasilkan perhitungan berikut.

## Perancangan Pengujian Algoritme

Pada perancangan untuk pengujian yang dilakukan pada algoritma *Neighbor Weight K-Nearest Neighbor* (NWKNN) bertujuan untuk mengetahui nilai parameter yang optimal dalam pemberian rekomendasi tema judul skripsi yang terkait bagi mahasiswa. Perancangan yang digunakan dalam pengujian nilai parameter pada algoritma NWKNN sebagai berikut.

1. Pengujian banyak generasi dan fitur.
2. Pengujian persentase banyak data *training* dan data *testing*.
3. Pengujian banyak nilai *K.*
4. Pengujian pengaruh nilai *K* dan *E*.

Dari skenario pengujian sebelumnya, proses pengujian dilakukan secara berulang ulang berdasarkan nilai parameter yang terbaik yang didapatkan sebelumnya.

### Pengujian Persentase Banyak Data *Training* dan Data *Testing*

Pada bagian pengujian ini dilakukan perbandingan persentase antara banyak data *training* dan data *testing* menggunakan fitur data yang optimum sebelumnya. Pengujian dimaksud untuk mengetahui nilai dari pengaruh persentase perbandingan data yang menghasilkan akurasi tertinggi. Pengujian perbandingan banyak data dilakukan dengan sebanyak 8 variasi yaitu dengan perbandingan data *Training*: data *Testing* sebesar 90%:10%, 80%:20%, 70%:30%, 60%:40%, 50%:50%, 40%:60%, 30%:70%, 20%:80%. Perancangan pengujian ini dapat dilihat pada Tabel 4.25.

Tabel 4.26 Perancangan Pengujian Presentasi Banyak Data *Training* dan Data *Testing*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Percobaa  Ke- | Akurasi Perbandingan Data Latih : Data Uji | | | | | | | |
| 20:80 | 30:80 | 40:60 | 50:50 | 60:40 | 70:30 | 80:20 | 90:10 |
| 1 |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |

### Pengujian Banyak Generasi dan Fitur

Pada pengujian generasi ini dilakukan untuk mengetahui nilai dari generasi terbaik mengenghasilkan nilai dan fitur yang optimal. Nilai yang digunakan dalam parameter banyak generasi adalah 10 – 100. Dari nilai tersebut diketahui apakah pengujian pada banyak generasi dapat mempengaruhi nilai dan fitur yang digunakan untuk menghasilkan akurasi optimal. Nilai dan fitur yang digunakan sebagai bahan perhitungan untuk proses klasifikasi selanjutnya. Perancangan pengujian banyak generasi dan fitur dapat dilihat pada bagian Tabel 4.26.

Tabel 4.27 Perancangan Pengujian Banyak Generasi Dan Fitur

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Fitur | Generasi Ke- | | | | | | | | | |
| 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| 1 |  |  |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |  |  |  |  |
| 16 |  |  |  |  |  |  |  |  |  |  |
| 17 |  |  |  |  |  |  |  |  |  |  |
| 18 |  |  |  |  |  |  |  |  |  |  |
| 19 |  |  |  |  |  |  |  |  |  |  |
| 20 |  |  |  |  |  |  |  |  |  |  |
| 21 |  |  |  |  |  |  |  |  |  |  |
| 22 |  |  |  |  |  |  |  |  |  |  |
| 23 |  |  |  |  |  |  |  |  |  |  |
| 24 |  |  |  |  |  |  |  |  |  |  |
| 25 |  |  |  |  |  |  |  |  |  |  |
| 26 |  |  |  |  |  |  |  |  |  |  |
| 27 |  |  |  |  |  |  |  |  |  |  |
| 28 |  |  |  |  |  |  |  |  |  |  |
| 29 |  |  |  |  |  |  |  |  |  |  |
| 30 |  |  |  |  |  |  |  |  |  |  |
| 31 |  |  |  |  |  |  |  |  |  |  |
| 32 |  |  |  |  |  |  |  |  |  |  |
| 33 |  |  |  |  |  |  |  |  |  |  |
| *Fitness* |  |  |  |  |  |  |  |  |  |  |

### Pengujian Banyak Nilai K

Pada bagian pengujian ini dilakukan menggunakan banyak data sebelumnya yang optimum untuk mengetahui pengaruh yang dihasilkan dari penggunaan nilai banyaknya nilai *K* terhadap nilai akurasi yang didapat sehingga menghasilkan akurasi yang optimal. Nilai tersebut merupakan nilai yang dijadikan input pada parameter *K*. Perancangan pengujian untuk mengetahui nilai *K* yang optimal terdapat pada bagian Tabel 4.27.

Tabel 4.28 Perancangan Pengujian bayak Nilai *K*

|  |  |  |  |
| --- | --- | --- | --- |
| Nilai *K* | Jumlah Benar | Jumlah Salah | Akurasi |
| 3 |  |  |  |
| 4 |  |  |  |
| 5 |  |  |  |
| 6 |  |  |  |
| 7 |  |  |  |
| 8 |  |  |  |
| 9 |  |  |  |

### Pengujian Pengaruh Nilai *K* dan *E*

Pada bagian pengujian ini dilakukan untuk mengetahui pengaruh nilai *K* dan *E* yang optimum sehingga menghasilkan akurasi yang optimum. Nilai tersebut dijadikan input dan parameter K dan E yang digunakan untuk klasifikasi akhir. Perancangan pengujian untuk mengetahui nilai *K* dan *E* yang optimum terdapat pada bagian Tabel 4.25.

Table 4.25 Pengaruh Nilai *K dan E* Terhadap Akurasi

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *K* | Akurasi *E* | | | | | | | |
| 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 3 |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |

# IMPLEMENTASI

## Implementasi Sistem

Pada bagian ini dilakukan penerapan dari perancangan sesuai dengan yang telah dibuat sebelumnya di rancangan algoritma *Neighbor Weight K-Nearest Neighbor* (NWKNN) dan seleksi fitur *Particle Swarm Optimization* (PSO). Dalam implementasi ini dibagi menjadi beberapa proses implementasi menjadi kode program untuk *pre-processing*, kode program normalisasi data, kode program penyeleksian situr, kode program perhitungan jarak, kode program perhitungan bobot, kode program perhitungan skor. Pada setiap proses memiliki sub proses kode program sesuai dengan perancangan algoritma.

### Implementasi *Pre-Processing Data*

Pada bagian implementasi pre-processing data merupakan proses untuk membaca data berupa *Excel* dengan extensi (.xls) dan mengubah datanya kedalam bentuk array *String* 2 Dimensi. Kode program untuk implementasi fungsi *pre-processing* data terdapat pada Kode Program 5.1.

|  |  |
| --- | --- |
| Kode Program : Fungsi *Pre-Processing* Data | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | public static String[][] data(File f) throws Exception {  Workbook wb = Workbook.getWorkbook(f);  Sheet s = wb.getSheet(0);  int row = s.getRows();  int col = s.getColumns();  String[][] hsl = new String[row][col];  for (int i = 0; i < row; i++) {  for (int j = 0; j < col; j++) {  Cell c = s.getCell(j, i);  hsl[i][j] = c.getContents();  }  }  return hsl;  } |

Kode Program 5.1 *Pre-Processing* Data

Penjelasan untuk implementasi Kode Program 5.1 dijabarkan sebagai berikut:

1. Baris 1 menjelaskan deklarasi fungsi untuk mengambil data dengan kembalian fungsi berupa array *String* 2 dimensi dengan parameter tipe data *File*.
2. Baris 2-5 menjelaskan inisialisasi variabel untuk mengambil *file excel* dari alamat yang tersimpan di variabel f, selanjutnya mengambil sheet, baris dan kolom yang digunakan.
3. Baris 6 menjelaskan pembuatan *array String* 2 dimensi sebagai tempat pengisian nilai dari *file excel* yang diambil.
4. Baris 8-11 menjelaskan proses pemindahan isi *file excel* kedalam array hsl untuk dilakukan perhitungan.
5. Baris 14 menjelaskan keluaran dari fungsi berupa *array* hsl.

### Implementasi Normalisasi data

Pada bagian implementasi normalisasi data bertujuan untuk mengubah data menggunakan nilai minimum dan maksimum untuk diubah dalam rentang nilai antara [0, 1]. Untuk implementasi fungsi terdapat pada Kode Program 5.2.

|  |  |
| --- | --- |
| Kode Program : Fungsi Normalisasi Data | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | public static String[][] normalisasi(String[][] x) {  String[][] hsl = new String[x.length][x[0].length];  double max = 0, min = 4;  for (int i = 0; i < x.length; i++) {  for (int j = 0; j < hsl[i].length - 2; j++) {  if (Double.parseDouble(x[i][j]) > max) {  max = Double.parseDouble(x[i][j]);  }  if (Double.parseDouble(x[i][j]) < min) {  min = Double.parseDouble(x[i][j]);  }  }  }  for (int i = 0; i < x.length; i++) {  for (int j = 0; j < hsl[i].length; j++) {  if (j < hsl[i].length - 2) {  double a = (Double.parseDouble(x[i][j]) - min)   / (max - min);  DecimalFormat df = new DecimalFormat("#.###");  hsl[i][j] = df.format(a);  } else {  hsl[i][j] = x[i][j];  }  }  }  return hsl; |

Kode Program 5.2 Normalisasi Data

Penjelasan untuk implementasi Kode Program 5.2 dijabarkan sebagai berikut:

1. Baris 1 menjelaskan deklarasi fungsi untuk normalisasi data dengan kembalian fungsi berupa array *String* 2 dimensi dengan parameter tipe data *array string* 2 dimensi.
2. Baris 2-3 menjelaskan inisialisasi variable *array string* 2 dimensi hsl sebagai tempat penyimpanan hasil normalisasi dan inisialisasi awal variable min dan max untuk tempat penyimpanan nilai minimum dan maximum.
3. Baris 4-13 menjelaskan proses pencarian nilai minimum dan maximum dari seluruh data yang ada.
4. Baris 15-25 menjelaskan proses normalisasi data mentah sehingga menghasilkan nilai antara [0, 1] dengan format 3 angka di belakang koma.
5. Baris 26 menjelaskan keluaran dari fungsi berupa *array* hsl.

### Implementasi Penyeleksian Fitur (PSO)

Pada bagian implementasi penyeleksian fitur (PSO) bertujuan untuk menyeleksi fitur-fitur yang nantinya digunakan sebagai bahan perhitungan sehingga menghasilkan nilai optimum dengan fitur yang minimum. Untuk implementasi dari fungsi implementasi penyeleksian fitur terdapat pada Kode Program 5.3.

|  |  |
| --- | --- |
| Kode Program : Fungsi Penyeleksian Fitur (PSO) | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37 | public static int[] pso(String[][] data, int nPartikel, int k, int e, double persen) {  int[][] partikel;  int generasi = 10;  double w = 0.5;  double c1 = 1;  double c2 = 1;  double[] fit = new double[nPartikel];  partikel = inisialAwal(data, nPartikel);  double[][] v = new double[nPartikel][data[0].length - 2];  int[][] pbest = new int[nPartikel][data.length - 2];  int[] gbest = new int[data.length - 2];  for (int i = 0; i < generasi; i++) {  for (int j = 0; j < nPartikel; j++) {  fit[j] = fitness(data, k, e, partikel[j], persen);  }  if (i == 0) {  Pbest = partikel;  pbest = Pbest;  fitPbest = fit;  gbest(partikel, fit);  gbest = Gbest;  } else {  updatePbest(partikel, pbest, fit, fitPbest);  pbest = Pbest;  gbest(Pbest, fitPbest);  gbest = Gbest;  double[][] temp = kecepatan(w, c1, c2, partikel,    Pbest, Gbest, v);  v = temp;  int[][] tempPosisi = updatePosisi(partikel, v);  partikel = tempPosisi;  }  }  return gbest;  } |

Kode Program 5.3 Penyeleksian Fitur (PSO)

Penjelasan untuk implementasi Kode Program 5.3 dijabarkan sebagai berikut:

1. Baris 1 menjelaskan deklarasi fungsi untuk penyeleksian fitur dengan kembalian fungsi berupa array *int* 1 dimensi dengan parameter tipe data *array string* 2 dimensi, banyak partikel tipe data integer, nilai k dan e bertipe data integer dan nilai persentase data yang digunakan bertipe data double.
2. Baris 3-12 menjelaskan inisialisasi variabel partikel, banyak generasi, nilai w, nilai c1, nilai c2, nilai fitness partikel, inisilisasi awal pembuatan partikel secara random, inisialisasi kecepan partikel, inisialisasi partikel Pbest, inisialisasi partikel Gbest.
3. Baris 14-17 menjelaskan proses pencarian nilai *fitness* dari setiap pertikel yang ada yang disimpan kedalam variabel fit.
4. Baris 19-23 menjelaskan proses insialisasi awal untuk partikel pbest, fitness dari Pbest, dan nilai partikel gbest.
5. Baris 24-35 menjelaskan proses pengupdetan nilai pbest, nilai gbest, dan perubahan posisi nilai partikel.
6. Baris 36 menjelaskan keluaran dari fungsi berupa *array* gbest.

### Implementasi Perhitungan Jarak

Pada bagian implementasi perhitungan jarak bertujuan untuk menghitung jarak antara data latih dengan data test yang nantinya digunakan sebagai bahan perhitungan selanjutnya. Untuk implementasi dari fungsi implementasi penyeleksian fitur terdapat pada Kode Program 5.4.

|  |  |
| --- | --- |
| Kode Program : Fungsi Perhitungan Jarak | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | public static double[][] jarak(String[][] x, String[] y) {  double[][] jarak = new double[x.length][2];  DecimalFormat df = new DecimalFormat("#.###");  int kelas = x[0].length - 1;  for (int i = 0; i < jarak.length; i++) {  double htng = 0;  for (int j = 0; j < x[i].length - 2; j++) {  double a = Math.pow(Double.parseDouble(x[i][j]) -   Double.parseDouble(y[j]), 2);  htng += a;  }  String a = df.format(Math.sqrt(htng));  jarak[i][0] = Double.parseDouble(a);  jarak[i][1] = Double.parseDouble(x[i][kelas]);  }  return jarak;  } |

Kode Program 5.4 Perhitungan Jarak

Penjelasan untuk implementasi Kode Program 5.4 dijabarkan sebagai berikut:

1. Baris 1 menjelaskan deklarasi fungsi untuk perhitungan jarak dengan kembalian fungsi berupa array *double* 2 dimensi dengan parameter tipe data *array string* 2 dimensi dan *array string* 1 dimensi.
2. Baris 2-4 menjelaskan menjelaskan inisialisasi variabel jarak yang berisi nilai jarak tiap-tiap data latih dengan data test, inisialisasi buat format nilai jarak dengan format 3 angka dibelakang koma dan inisialisasi kelas dari tiap jarak yang didapat.
3. Baris 5-13 menjelaskan proses perhitungan jarak dari data tes dengan tiap-tiap data latih dengan format 3 angka di belakang koma.
4. Baris 14-15 menjelaskan proses pengisian nilai *array* jarak yang berisi nilai jarak dengan kelasnya.
5. Baris 17 menjelaskan keluaran dari fungsi berupa *array* gbest.

### Implementasi Perhitungan Bobot

Pada bagian implementasi perhitungan bobot bertujuan untuk menghitung bobot untuk tiap kelas yang tersedia. Untuk implementasi dari fungsi implementasi penyeleksian fitur terdapat pada Kode Program 5.5.

|  |  |
| --- | --- |
| Kode Program : Fungsi Perhitungan Bobot | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35 | public static double[][] bobot(double[][] jarak, int k, int e) {  double[][] hsl = new double[3][2];  double[] kumpulan = new double[3];  double min = k;  for (int j = 0; j < k; j++) {  if (jarak[j][1] == 1.0) {  kumpulan[0] += 1;  } else if (jarak[j][1] == 2.0) {  kumpulan[1] += 1;  } else if (jarak[j][1] == 3.0) {  kumpulan[2] += 1;  }  }  for (int i = 0; i < kumpulan.length; i++) {  if (kumpulan[i] != 0) {  if (kumpulan[i] < min) {  min = kumpulan[i];  }  }  }  for (int i = 0; i < hsl.length; i++) {  if (kumpulan[i] > 0) {  double b = e;  double a = Math.pow(kumpulan[i] / min, (1 / b));  hsl[i][0] = (1 / a);  hsl[i][1] = kumpulan[i];  } else {  hsl[i][0] = 0;  hsl[i][1] = kumpulan[i];  }  }  return hsl;  } |

Kode Program 5.5 Perhitungan Bobot

Penjelasan untuk implementasi Kode Program 5.5 dijabarkan berikut:

Baris 1 menjelaskan deklarasi fungsi untuk perhitungan bobot dengan kembalian fungsi berupa array *double* 2 dimensi dengan parameter tipe data *array double* 2 dimensi, nilai k dan e bertipe data integer.

1. Baris 2-4 menjelaskan proses inisialisasi variabel hsl sebagai tempat penyimpanan bobot, variabel kumpulan sebagai tempat penyimpanan banyak anggota dari tiap kelas dan variabel min sebagai nilai k untuk mengambil k data dari urutan paling atas.
2. Baris 5-11 menjelaskan proses penghitungan jumlah anggota dari tiap kelas yang tersaring dari nilai k.
3. Baris 15-18 menjelaskan proses mencari anggota paling sedikit dari keseluruhan kelas yang bukan bernilai 0.
4. Baris 23-31 menjelaskan proses menghitung nilai bobot untuk tiap kelas.
5. Baris 34 menjelaskan keluaran dari fungsi berupa *array* hsl.

### Implementasi Perhitungan Skor

Pada bagian implementasi perhitungan skor bertujuan untuk menghitung skor akhir untuk tiap kelas yang tersedia. Untuk implementasi dari fungsi implementasi penyeleksian fitur terdapat pada Kode Program 5.6.

|  |  |
| --- | --- |
| Kode Program : Fungsi Perhitungan Skor | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | public static int skor(double[][] jarak, double[][] bobot, int k) {  double[][] hslSkor = new double[bobot.length][2];  int kelas;  for (int i = 0; i < hslSkor.length; i++) {  double count = 0;  for (int j = 0; j < k; j++) {  if (i + 1 == jarak[j][1]) {  if (bobot[i][1] > 0) {  double a = jarak[j][0];  count += a;  } else {  count = 0;  }  }  }  hslSkor[i][0] = bobot[i][0] \* count;  hslSkor[i][1] = i + 1;  }  kelas = Max(hslSkor);  return kelas;  } |

Kode Program 5.6 Perhitungan Skor

Penjelasan untuk implementasi Kode Program 5.6 dijabarkan sebagai berikut:

1. Baris 1 menjelaskan deklarasi fungsi untuk perhitungan skor dengan kembalian fungsi berupa integer dengan parameter tipe data *array double* 2 dimensi untuk jarak dan tipe data *array double* 2 dimensi untuk bobot, nilai k.
2. Baris 2-3 menjelaskan inisialisasi variabel hslSkor sebagai tempat penampungan nilai dari skor dan inisialisasi variabel kelas sebagai tempat penyimpanan hasil prediksi kelas dari sistem.
3. Baris 4-17 menjelaskan proses pencarian nilai skor untuk tiap-tiap kelas dan menyimpanya kedalam *array* hslSkor.
4. Baris 19 menjelaskan proses pencarian prediksi kelas dengan mencari nilai skor tertinggi dari keseluruhan kelas.
5. Baris 20 menjelaskan keluaran dari fungsi berupa *integer* kelas.

# HASIL DAN PEMBAHASAN

Pada bagian ini dilakukan penjelasan terhadap hasil dari pengujian implementasi sistem sebelumnya. Hasil pengujian dilakukan berdasarkan nilai akurasi yang diperoleh dari sistem. Hasil dan pembahasan mengenai pengujian yang dilakukan mengenai pengujian banyak data, pengujian banyak generasi dan fitur, pengujian nilai K, dan pengujian pengaruh nilai *K* dan *E*. Setiap hasil pengujian yang diperoleh akan dilakukan pembahasan.

## Hasil Pengujian

### Pengujian banyak data

Pada bagian ini merupakan hasil pengujian perbandingan persentase antara banyak data *Training* dan data *Testing*. Pengujian ini dilakukan untuk mengetahui nilai perbandingan data yang menghasilkan akurasi tertinggi. Pengujian ini dilakukan sebanyak 8 variasi yaitu dengan perbandingan data *Training*: data *Testing* sebesar 90%:10%, 80%:20%, 70%:30%, 60%:40%, 50%:50%, 40%:60%, 30%:70%, 20%:80%. Menggunkan nilai *K* dan *E* awal sebesar 3 dan 2 dengan perulangan sebanyak 5 kali. Hasil dari pengujian dapat dilihat pada Tabel 6.1.

Tabel 6.1 Hasil Pengujian Persentase Banyak Data *Training* dan Data *Testing*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Percobaa  Ke- | Akurasi Perbandingan Data Latih : Data Uji | | | | | | | |
| 20:80 | 30:80 | 40:60 | 50:50 | 60:40 | 70:30 | 80:20 | 90:10 |
| 1 | 53,73% | 57,72% | 56% | 56,62% | 56,71% | 60% | 63,63% | 76,47% |
| 2 | 55,23% | 56,41% | 55% | 54,21% | 58,20% | 64% | 66,66% | 70,58% |
| 3 | 55,22% | 54,70% | 58% | 59,03% | 58,20% | 62% | 66,66% | 76,47% |
| 4 | 54,47% | 55,55% | 59% | 57,83% | 55,22% | 62% | 69,69% | 82,35% |
| 5 | 54,47% | 58,11% | 57% | 55,42% | 56,71% | 62% | 63,63% | 88,23% |

Dari hasil pada Tabel 6.1 dapat ditampilkan nilai akurasi dari 5 kali percobaan yang ditampilkan dalam bentuk diagram garis, Tabel 6.1 dapat diketahui bahwa nilai akurasi mulai semakin optimal mulai dari perbandingan 40:60 menuju nilai paling optimal yang terdapat pada perbandingan 90:10. Dan dapat dilihat nilai tersebut terus mengalami kenaikan seiring bertambahnya data latih, nilai akurasi mengalami peningkatan yang signifikan pada perbandingan 90:10 di setiap percobaannya. Untuk melihat perkembangan nilai dalam bentuk grafik garis yang dapat dilihat pada Gambar 6.1.

Gambar 6.1 Hasil Pengujian Persentase Banyak Data *Training* dan Data *Testing*

Dari hasil grafik Gambar 6.1 dan Tabel 6.1 dapat dilihat bahwa perbandingan data *training* dengan data *testing* yaitu 90%: 10% memiliki akurasi tertinggi dibandingkan dengan perbandingan data lainnya, sehingga dengan hasil grafik diatas maka nilai perbandingan yang digunakan sebagai bahan pengujian selanjutnya yaitu perbandingan 90% data *training* dan 10% data *testing*.

### Pengujian Banyak Generasi dan Fitur

Pada bagian ini merupakan hasil pengujian generasi untuk mengetahui nilai dari generasi terbaik mengenghasilkan nilai dan fitur yang optimal. Nilai yang digunakan dalam parameter banyak generasi adalah 10 – 100. Dari nilai tersebut diketahui apakah pengujian pada banyak generasi dapat mempengaruhi nilai dan fitur yang digunakan untuk menghasilkan akurasi optimal. Nilai dan fitur yang digunakan sebagai bahan perhitungan untuk proses klasifikasi selanjutnya. Perancangan pengujian banyak generasi dan fitur dapat dilihat pada bagian Tabel 6.2.

Tabel 6.2 Hasil Pengujian Banyak Generasi dan Fitur

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Matkul | Generasi Ke- | | | | | | | | | |
| 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 3 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 7 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 12 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 13 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 14 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 15 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 16 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 17 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 20 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 21 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 23 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 24 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 25 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 26 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 27 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 28 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 29 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 30 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 31 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 32 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 33 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| *Fitness* | 0.52 | 0.64 | 0.76 | 0.76 | 0.88 | 0.88 | 0.88 | 0.88 | 0.88 | 0.88 |

Dari hasil Tabel 6.2 dapat ditampilkan nilai akurasi yang dihasilkan dari 10- 100 generasi yang dilakukan dalam bentuk diagram garis pada Gambar 6.2.

Gambar 6.2 Hasil Pengujian Generasi dan Fitur

Dari hasil grafik pada Gambar 6.2 dan Tabel 6.2 dapat dilihat bahwa *fitness optimal* mulai dicapai pada generasi ke-50 dengan akurasi sebesar 0.88, dan dari fitness terbesar tesebut diketahui mata kuliah yang dipakai berdasarkan Tabel 6.2 yaitu : MK2, MK6, MK7, MK11, MK12, MK13, MK14, MK15, MK21, MK24, MK26, MK27, MK28, MK31, MK32. Mata kuliah yang diperoleh kemudian digunakan sebagai parameter mata kuliah yang digunkan untuk proses klasifikasi selanjutnya.

### Pengujian Nilai K

Pada bagian ini merupakan hasil pengujian untuk mengetahui pengaruh yang dihasilkan dari penggunaan nilai banyaknya nilai *K* terhadap nilai akurasi yang di dapat sehingga menghasilkan akurasi yang optimal. Nilai tersebut merupakan nilai yang dijadikan input pada parameter *K*. Perancangan pengujian untuk mengetahui nilai *K* yang optimal terdapat pada bagian Tabel 6.3.

Tabel 6.3 Hasil Pengujian Nilai K

|  |  |  |  |
| --- | --- | --- | --- |
| Nilai *K* | Jumlah Benar | Jumlah Salah | Akurasi |
| 3 | 15 | 2 | 88,23% |
| 4 | 11 | 6 | 64,7% |
| 5 | 10 | 7 | 58,82% |
| 6 | 13 | 4 | 76,47% |
| 7 | 11 | 6 | 64,7% |
| 8 | 10 | 7 | 58,82% |
| 9 | 9 | 8 | 52,94% |

Dari hasil Tabel 6.3 dapat ditampilkan nilai akurasi yang dihasilkan oleh nilai *K* yang berbeda dalam bentuk diagram garis pada gambar 6.3.

Gambar 6.3 Hasil Pengujian Nilai K

Berdasarkan hasil dari grafik Gambar 6.3 dan Tabel 6.3 dapat diketahui bahwa nilai K yang optimum untuk mendapatkan akurasi tertinggi pada nilai K = 3 dengan akurasi 88,23% sehingga dengan ini nilai K optimum digunakan untuk proses selanjutnya.

### Pengujian Pengaruh Nilai K dan E

Pada bagian ini merupakan hasil pengujian dari pengaruh nilai *K* dan *E* yang optimum sehingga menghasilkan akurasi yang optimum. Nilai tersebut dijadikan input dan parameter K dan E yang digunakan untuk klasifikasi akhir. Perancangan pengujian untuk mengetahui nilai *K* dan *E* yang optimum terdapat pada bagian Tabel 6.4.

Tabel 6.4 Hasil Pengujian Pengaruh Nilai K dan E

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *K* | Akurasi *E* | | | | | | | |
| 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 3 | 88.23% | 88.23% | 88.23% | 88.23% | 88.23% | 88.23% | 88.23% | 88.23% |
| 4 | 64.70% | 64.70% | 64.70% | 64.70% | 64.70% | 64.70% | 64.70% | 64.70% |
| 5 | 58.82% | 58.82% | 58.82% | 58.82% | 58.82% | 58.82% | 58.82% | 58.82% |
| 6 | 76.47% | 76.47% | 76.47% | 76.47% | 76.47% | 76.47% | 76.47% | 76.47% |
| 7 | 64.70% | 64.70% | 64.70% | 64.70% | 64.70% | 64.70% | 64.70% | 64.70% |
| 8 | 58.82% | 58.82% | 58.82% | 58.82% | 58.82% | 58.82% | 58.82% | 58.82% |
| 9 | 52.94% | 58.82% | 58.82% | 58.82% | 58.82% | 58.82% | 58.82% | 58.82% |

Dari hasil Tabel 6.4 dapat ditampilkan nilai akurasi yang dihasilkan oleh nilai *K* dengan nilai *E* yang berbeda dalam bentuk diagram garis pada Gambar 6.4.

Gambar 6.4 Hasil Pengujian Pengaruh NIlai K dan E

Berdasarkan hasil dari grafik Gambar 6.4 dan Tabel 6.4 nilai K dan E yang optimum bernilai 3 dan 2, pengaruh nilai E pada penelitian ini masih kurang signifikan di yang dapat dilihat nilai E hanya menambah akurasi pada K = 9 dan pada K yang lainnya tidak ada megalami perubahan

# PENUTUP

Pada bagian penutup menjelaskan dari kesimpulan dan saran dari penelitian rekomendasi pengambilan tema judul skripsi menggunakan *PSO-Neighbor Weighted K-Nearest Neighbor* (NWKNN). Bagian kesimpulan menjelaskan hasil jawaban dari rumusan masalah pada penelitian ini. Bagian saran menjelaskan kekurangan dalam penelitian ini yang dapat digunkan sebagai bahan referensi untuk penelitian selanjutnya.

## Kesimpulan

Berdasarkan hasil dan pembahasan pada Bab sebelumnya tentang rekomendasi pengambilan tema judul skripsi menggunakan *PSO-Neighbor Weighted K-Nearest Neighbor* (NWKNN). Kesimpulan yang diperoleh sebagai berikut.

1. Algoritma *PSO-Neighbor Weighted K-Nearest Neighbor* (NWKNN) untuk rekomendasi pengambilan tema judul skripsi menghasilkan nilai parameter yang optimal melalui pengujian persentase data, banyak fitur nilai K dan hubungan nilai K dan E. Pada pengujian persentase banyak perbandingan data *Training* dan data *Testing* yang digunakan sebesar 90% : 10% dengan menghasil nilai akurasi tertinggi rata-rata 76.47%. Pada pengujian generasi dan fitur menghasilkan generasi yang mulai konstan pada generasi ke 50 dan dengan mata kuliah yang digunakan sebanyak 15 mata kuliah yaitu : MK2, MK6, MK7, MK11, MK12, MK13, MK14, MK15, MK21, MK24, MK26, MK27, MK28, MK31, MK32. Pada pengujian nilai K diperoleh nilai K optimum sebesar 3. Pada pengujian nilai K dan E diperoleh nilai K dan E optimum sebesar 3 dan 2.
2. Algoritma *PSO-Neighbor Weighted K-Nearest Neighbor* (NWKNN) untuk rekomendasi pengambilan tema judul skripsi menghasilkan nilai optimum menggunakan parameter sebelumnya diperoleh hasil akurasi sebesar 88.28%.

## Saran

Saran yang digunakan untuk penelitian selanjutnya sebagai berikut:

1. Peningkatan pada proses pengelompokan/*clustering* pada data yang dimana peneliti masih menggunkan K-Means dengan K = 3 dan orentasi pengelompokan berdasarkan kemiripan judul skripsi sehingga nantinya dapat menambah akurasi pada proses klasifikasi.
2. Pengoptimalan dalam seleksi fitur *PSO* sehingga mendapatkan hasil optimal yang lebih cepat dan tidak tejebak pada *local optimum*.

DAFTAR REFERENSI

Anshori, L., Putri, R. R. M. & T., 2018. Implementasi Metode K-Nearest Neighbor untuk Rekomendasi Keminatan Studi (Studi Kasus: Jurusan Teknik Informatika Universitas Brawijaya). *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 2(7), pp. 2745-2753.

Feizar, F. H., I. & Yudistira, N., 2014. Analisis Sentimen Opini Film Berbahasa Indonesia Berbasis Kamus Menggunakan Metode Neighbor-Weighted K-Nearest Neighbor. *Repositori Jurnal Mahasiswa PTIIK UB,* 3(11).

Hadi, A. H., Ratnawati, D. E. & Dewi, C., 2018. Identifikasi Penyakit Gagal Ginjal Menggunakan Metode Neighbor Weighted K-Nearest Neighbor (NWKNN). *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 2(9), pp. 2562-2569.

Han, J., Kamber, M. & Pei, J., 2011. *Data Mining Concepts and Technique,* Massachusetts: Elsevier,Inc.

Indriati & Ridok, A., 2016. Sentiment Analysis For Review Mobile Applications Using Neighbor Method Weighted K-Nearest Neighbor (Nwknn). *Journal of Environmental Engineering & Sustainable Technology,* 3(1), pp. 23-32.

Khairunnisa, 2017. *Pengaruh Mata Kuliah Bimbingan Penulisan Skripsi Terhadap Kemampuan Menulis Karya Ilmiah Oleh Mahasiswa/I Ilmu Perpustakaan Fakultas Adab Dan Humaniora Tahun Akademik 2011/2012,* Banda Aceh: UIN AR-RANIRY DARUSSALAM.

Kunwar, S., 2013. *Code Project For Those Who Code.* [Online]   
Available at: https://www.codeproject.com/Articles/439890/Text-Documents-Clustering-using-K-Means-Algorithm  
[Accessed 18 August 2020].

Prasetyo, E., 2014. *Data Mining, Mengolah Data Menjadi Informasi Menggunakan Matlab.* 1st ed. Yogyakarta: AndiPublisher.

RIADI, M., 2017. *Pengertian, Fungsi, Proses, dan Tahapan Data Mining..* [Online]   
Available at: https://www.kajianpustaka.com/2017/09/data-mining.html.  
[Accessed 9 Mei 2017].

Ridok, A. & Latifah, R., 2015. Klasifikasi Teks Bahasa Indonesia Pada Corpus Tak Seimbang Menggunakan NWKNN. *Konferensi Nasional Sistem & Informatika 2015 STMIK STIKOM Bali,* pp. 222-227.

Santosa, B., 2007. *Teknik Pemanfaatan Data untuk Keperluan Bisnis.* Yogyakarta: Graha Ilmu.

Satria, A., Marji & Ratnawati, D. E., 2019. Klasifikasi Jenis Kanker Berdasarkan Struktur Protein Menggunakan Metode Neighbor Weighted K-Nearest Neighbor (NWKNN). *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 3(4), pp. 3617-3624.

Somantri, O., Wiyono, S. & D., 2016. Metode K-Means untuk Optimasi Klasifikasi Tema Tugas Akhir Mahasiswa Menggunakan Support Vector Machine (SVM). *Scientific Journal of Informatics,* 3(1).

Tan, S., 2005. Neighbor-weighted K-nearest. *Expert Systems with Applications,* pp. 667-671.

UB, F. I. K., 2018. *Buku Panduan Skripsi Fakultas Ilmu Komputer.* KETIGA ed. Malang: Fakultas Ilmu Komputer UB.

UB, P. F., 2017. *Fakultas Ilmu Komputer Universitas Brawijaya.* [Online]   
Available at: https://filkom.ub.ac.id/  
[Accessed 1260 54391 2019].

Widarto, 2017. Faktor Penghambat Studi Mahasiswa Yang Tidak Lulus Tepat Waktu Di Jurusan Pendidikan Teknik Mesin Ft Uny. *Jurnal Dinamika Vokasional Teknik Mesin,* 2(2548-7590), pp. 127-138.

LAMPIRAN A DATA

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MK1 | MK2 | MK3 | MK4 | MK5 | MK6 | MK7 | MK32 | MK33 |
| C | C | C | C | B | B | B | C | C |
| C | B | C | C | A | A | A | B | A |
| B | C | C | C | C | B | C | C | B |
| B | B | C | C | C | B | B | C | C |
| C | B | A | B | B | B | B | B | C |
| C | C | A | C | C | B | B | C | C |
| B | B | C | C | C | B | C | C | C |
| B | B | C | A | B | B | C | B | A |
| B | C | B | C | C | B | A | C | C |
| B | B | C | C | C | B | A | B | A |
| B | B | C | C | C | B | C | C | C |
| C | B | B | C | A | C | B | B | B |
| C | B | B | C | C | B | A | B | C |
| B | B | B | B | C | B | A | C | B |
| B | B | A | B | C | B | A | C | C |
| B | B | B | C | C | B | B | C | C |
| B | B | B | A | C | B | A | B | B |
| C | B | B | C | C | B | A | B | B |
| B | A | B | B | C | A | B | B | C |
| B | C | C | B | C | B | A | C | C |
| B | B | B | A | B | A | A | B | B |

Data Selengkapnya: <https://github.com/rizkyyud/Lampiran.git>

LAMPIRAN B Daftar mata kuliah

|  |  |
| --- | --- |
| **Keterangan** | |
| **KODE** | **NAMA MATA KULIAH** |
| **MK1** | **Fisiologi Manusia** |
| **MK2** | **Anatomi Manusia** |
| **MK3** | **Renang** |
| **MK4** | **Senam** |
| **MK5** | **Adm. dan Organisasis Pertandingan Olahraga** |
| **MK6** | **Kebugaran Jasmani** |
| **MK7** | **Fisiologi Olahraga** |
| **MK8** | **Biokimia Olahraga** |
| **MK9** | **Sepak Bola** |
| **MK10** | **Manajemen Olahraga** |
| **MK11** | **Atletik** |
| **MK12** | **Kinesiologi Olahraga** |
| **MK13** | **Olahraga Lansia** |
| **MK14** | **Gizi Olahraga** |
| **MK15** | **Teori dan Metode latihan** |
| **MK16** | **Dasar-dasar Fitness** |
| **MK17** | **Farmakologi** |
| **MK18** | **Pencak Silat** |
| **MK19** | **Bola Voli** |
| **MK20** | **Cedera Olahraga** |
| **MK21** | **Perkembangan Motorik** |
| **MK22** | **Basket** |
| **MK23** | **Senam Aerobik** |
| **MK24** | **Wanita dan Olahraga** |
| **MK25** | **Kesehatan Olahraga** |
| **MK26** | **Tes, Pengukuran dan Evaluasi Olahraga** |
| **MK27** | **Massase Olahraga** |
| **MK28** | **Biomekanika Olahraga** |
| **MK29** | **Terapi Fisik Olahraga** |
| **MK30** | **Olahraga Adaptif** |
| **MK31** | **Massage Olahraga** |
| **MK32** | **Fisiologi II** |
| **MK33** | **Anatomi II** |