

**TUGAS AKHIR – KI003498**

**DIAGNOSIS PENYAKIT BERDASARKAN GEJALA MENGGUNAKAN *FREQUENT PATTERN TREE GROWTH***

**KEVIN ALIF FACHREZA**

**NRP 5114100128**

Dosen Pembimbing I

Dr. Eng. Chastine Fatichah, S.Kom., M.Kom

Dosen Pembimbing II

Anny Yuniarti, S.Kom, M.Comp.Sc

Departemen Informatika

Fakultas Teknologi Informasi dan Komunikasi

Institut Teknologi Sepuluh Nopember

Surabaya 2018

***(Halaman ini sengaja dikosongkan)***



**TUGAS AKHIR – KI003498**

**DIAGNOSIS PENYAKIT BERDASARKAN GEJALA MENGGUNAKAN *FREQUENT PATTERN TREE GROWTH***

**KEVIN ALIF FACHREZA**

**NRP 5114100128**

**Dosen Pembimbing I**

**Dr. Eng. Chastine Fatichah, S.Kom., M.Kom**

**Dosen Pembimbing II**

**Anny Yuniarti, S.Kom, M.Comp.Sc**

**Departemen Informatika**

**Fakultas Teknologi Informasi dan Komunikasi**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2018**

***(Halaman ini sengaja dikosongkan)***



**UNDERGRADUATE THESES – KI003498**

**DISEASE DIAGNOSIS BASED ON SYMPTOMS USING FREQUET FP TREE GROWTH**

**KEVIN ALIF FACHREZA**

**NRP 5114100128**

**First Advisor**

**Dr. Eng. Chastine Fatichah, S.Kom., M.Kom**

**Second Advisor**

**Anny Yuniarti, S.Kom, M.Comp.Sc**

**Department of Informatics**

**Faculty of Information Technology and Communication**

**Sepuluh Nopember Institute of Technology**

**Surabaya 2018**

***(Halaman ini sengaja dikosongkan)***

**LEMBAR PENGESAHAN**

**DIAGNOSIS PENYAKIT BERDASARKAN GEJALA MENGGUNAKAN FREQUENT PATTERN TREE GROWTH**

**TUGAS AKHIR**

Diajukan Untuk Memenuhi Salah Satu Syarat

Memperoleh Gelar Sarjana Komputer

pada

Bidang Studi Kecerdasan Komputasional

Program Studi S-1 Departemen Informatika

Fakultas Teknologi Informasi dan Komunikasi

Institut Teknologi Sepuluh Nopember

Oleh:

**KEVIN ALIF FACHREZA**

**NRP : 5114100128**

Disetujui oleh Pembimbing Tugas Akhir:

1. Dr. Eng. Chastine Fatichah, S.Kom., M.Kom. .....................

(NIP. 197512202001122002) (Pembimbing 1)

1. Anny Yuniarti, S.Kom, M.Comp.Sc ......................

(NIP. 198106222005012002) (Pembimbing 2)

**SURABAYA**

**JUNI, 2018**

***(Halaman ini sengaja dikosongkan)***

**DIAGNOSIS PENYAKIT BERDASARKAN GEJALA MENGGUNAKAN *FREQUENT PATTERN TREE GROWTH***

|  |  |  |
| --- | --- | --- |
| **Nama Mahasiswa** | **:** | **Kevin Alif Fachreza** |
| **NRP** | **:** | **5114100128** |
| **Departemen** | **:** | **Informatika FTIK-ITS** |
| **Dosen Pembimbing 1** | **:** | **Dr. Eng. Chastine Fatichah, S.Kom., M.Kom** |
| **Dosen Pembimbing 2** | **:** | **Anny Yuniarti, S.Kom, M.Comp.Sc** |

# Abstrak

*Machine learning* merupakan salah satu bidang di teknologi informasi yang sedang naik daun. Teknologi ini dimanfaatkan diberbagai bidang kehidupan manusia, mulai dari perbankan, transportasi, sosial media, termasuk kesehatan. Potensi pemanfaatan *machine learning* pada bidang kesehatan sangatlah besar. Salah satunya adalah diagnosis penyakit. Diagnosis penyakit atau dikenal dengan tahap anamnesis pada kedokteran adalah suatu proses dimana dokter akan menanyakan kepada pasien gejala gejala yang dialami pasien. Sehingga dokter dapat memperkecil kemungkinan penyakit yang mungkin dialami pasien dan melakukan tes penunjang seperti lab atau radiologi untuk mendapatkan keputusan diagnosis final.

Banyak aplikasi yang dapat mendiagnosis penyakit, akan tetapi *user experience* yang buruk menyebabkan diagnosis meleset. Aplikasi tersebut biasanya akan meminta pengguna untuk memasukkan gejala mereka baik teks maupun berupa *checkboxes*. Padahal pasien tidak terlalu mengetahui gejala gejala spesifik atau yang berkorelasi yang dapat membantu dokter dalam mendiagnosis penyakit secara signifikan.

Aplikasi yang dibuat pada tugas akhir ini akan berfokus pada bagaimana pasien dapat memasukkan gejala yang dialami dengan spesifik dan sejelas mungkin. Sehingga dokter dapat memberikan diagnosis yang lebih baik. Untuk menyelesaikan masalah tersebut, akan digunakan algoritma *Frequent Pattern Tree Growth*, yaitu salah satu metode *association rules* dimana algoritma ini dapat memetakan gejala gejala yang berkaitan, sehingga aplikasi dapat menanyakan gejala yang mungkin dialami oleh pasien berdasarkan gejala yang telah dimasukkan sebelumnya oleh pasien.

Dengan jumlah diagnosis penyakit sebanyak 66 penyakit, aplikasi ini memberikan hasil yang cukup baik. Dibuktikan dengan skenario pengujian dengan 198 data menghasilkan rata rata akurasi sebesar 69% dengan *classifier Naïve Bayes.* Dan rata rata akurasi sebesar 67% dengan *classifier SVM.* Aplikasi juga diujikan kepada dokter spesialis yang menghasilkan 80% hasil diagnosis relevan dengan gejala yang dimasukkan.

**Kata kunci: diagnosis, gejala, aplikasi**

**DISEASE DIAGNOSIS BASED ON SYMPTOMS USING FREQUENT PATTERN TREE GROWTH**

|  |  |  |
| --- | --- | --- |
| **Student’s Name** | **:** | **Kevin Alif Fachreza** |
| **Student’s ID** | **:** | **5114100128** |
| **Departemen** | **:** | **Informatika FTIK-ITS** |
| **First Advisor** | **:** | **Dr. Eng. Chastine Fatichah, S.Kom., M.Kom** |
| **Second Advisor** | **:** | **Anny Yuniarti, S.Kom, M.Comp.Sc** |

# Abstract

*Machine learning is currently one of the promising tech. This technology can be used in any fields, banking, transportation, social media, include health. Machine learning has a huge potency in health industry. One of them is diseases diagnosis. Diseases diagnosis or known as anamnesis in medical is a process where doctor will ask the patient about the symptoms or experience that patient felt. So doctor can narrow the possibility of diseases that the patient may suffer and do some lab and radiology tests to get final diagnosis decision*

*There’s a lot app that able to diagnose diseases, but with bad user experience made the diagnosis went fumble. Those apps will asks user to input their symptoms using text or checkboxes. Whereas patients don’t really know the specific symptoms or the symptoms that correlated to their main symptoms which can help doctor to diagnose disease significantly.*

*The app that made for this final project focused on how patient can input their symptoms specifically and clearly. So doctor can give better diagnosis decision. To solve that problem, Frequent Pattern Tree Growth algorithm will be used, Frequent Pattern Tree Growth is one of the association rules methods where this algorithm can map which symptoms is related to other symptoms, so the app can ask the next symptoms based on the previous symptoms that inputted by user.*

*With 66 diagnoses, this app gave a pretty good result. Proven by scenarios of testing with 198 datas, yielding 69% average accuracy with classifier Naïve Bayes. And 67% average accuracy with classifier SVM. This app also tested to a doctor specialist which came by a good result, 80% of the diagnosis result relevant to the inputted symptoms.*

***Keywords : diagnosis, symptoms, app***

# KATA PENGANTAR

Alhamdulillahirabbil’alamin, segala puji dan syukur bagi Allah SWT, yang telah melimpahkan rahmat, dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul **“DIAGNOSIS PENYAKIT BERDASARKAN GEJALA MENGGUNAKAN FREQUENT PATTERN TREE GROWTH”.**

Pengerjaan Tugas Akhir ini merupakan suatu kesempatan yang berharga bagi penulis. Dengan pengerjaan Tugas Akhir, penulis dapat memperdalam, meningkatkan, serta menerapkan apa yang telah didapatkan penulis selama menempuh perkuliahan di Teknik Informatika ITS.

Terselesaikannya Tugas Akhir ini tidak lepas dari bantuan dan dukungan dari berbagai pihak. Dan dalam kesempatan ini penulis mengucapkan rasa syukur dan terima kasih kepada:

1. Allah SWT, karena atas izin-Nya lah penulis dapat menyelesaikan Tugas Akhir dengan baik.
2. Keluarga penulis (Mama, Papa, Yangti, Yangkong, Rehun, dan keluarga penulis yang lain) yang selalu memberikan dukungan moral dan material yang tidak ternilai.
3. Ibu Dr.Eng. Chastine Fatichah, S.Kom, M.Kom selaku pembimbing I Tugas Akhir yang telah memberikan banyak waktu untuk berdiskusi dan memberi semangat dan motivasi kepada penulis untuk menyelesaikan Tugas Akhir.
4. Ibu Anny Yuniarti, S.Kom, M.Comp.Sc selaku pembimbing II Tugas Akhir yang telah memberikan banyak yang telah memberikan bimbingan dan dukungan selama penulis menyelesaikan Tugas Akhir.
5. Dr Adi Suriyanto, Sp.OT selaku dokter yang meluangkan waktu dan pikiran dalam memberikan masukan dan melakukan pengujian pada Tugas Akhir ini.
6. Annisa Rahmawati, S.KG yang memberikan dukungan moral, motivasi, masukan dan hiburan pada pengerjaan Tugas Akhir ini.
7. Teman teman Jojoran yang menyediakan informasi dan tempat serta fasilitas dalam mengerjakan Tugas Akhir ini.
8. Yang terakhir untuk orang orang yang membaca dan akan mengembangkan Tugas Akhir ini. Semoga dapat mewujudkan aplikasi yang diimpikan.

Penulis memohon maaf apabila terdapat kekurangan dalam penulisan Tugas Akhir ini. Kritik dan saran penulis harapkan untuk perbaikan dan pembelajaran di kemudian hari. Semoga Tugas Akhir ini dapat memberikan Manfaat yang sebesar besarnya.

Surabaya, 07 Juni 2018

Kevin A Fachreza

# DAFTAR ISI

[**LEMBAR PENGESAHAN** vii](#_Toc517457130)

[Abstrak ix](#_Toc517457131)

[Abstract xi](#_Toc517457132)

[KATA PENGANTAR xiii](#_Toc517457133)

[DAFTAR ISI xv](#_Toc517457134)

[DAFTAR GAMBAR xx](#_Toc517457135)

[DAFTAR TABEL xxii](#_Toc517457136)

[DAFTAR KODE SUMBER xxiv](#_Toc517457137)

[BAB I PENDAHULUAN 1](#_Toc517457138)

[1.1 Latar Belakang 1](#_Toc517457139)

[1.2 Rumusan Masalah 2](#_Toc517457140)

[1.3 Batasan Masalah 2](#_Toc517457141)

[1.4 Tujuan Pembuatan Tugas Akhir 2](#_Toc517457142)

[1.5 Manfaat Tugas Akhir 2](#_Toc517457143)

[1.6 Metodologi 3](#_Toc517457144)

[1.7 Sistematika Penulisan Laporan Tugas Akhir 4](#_Toc517457145)

[BAB II DASAR TEORI 7](#_Toc517457146)

[2.1 Anamnesis 7](#_Toc517457147)

[2.1.1 Pengertian Anamnesis 7](#_Toc517457148)

[2.1.2 Langkah Langkah Anamnesis 7](#_Toc517457149)

[2.2 Frequent Pattern Tree Growth 9](#_Toc517457150)

[2.2.1 Pengertian Frequent Pattern Tree Growth 9](#_Toc517457151)

[2.2.2 Support 9](#_Toc517457152)

[2.2.3 Confidence 10](#_Toc517457153)

[2.2.4 Algoritma FP Tree Growth 10](#_Toc517457154)

[2.4 Naïve Bayes 10](#_Toc517457155)

[2.5 *Support Vector Machine* 11](#_Toc517457156)

[2.6 *Scikit Learn* 11](#_Toc517457157)

[2.7 *Python – Flask* 11](#_Toc517457158)

[2.8 *PHP - Laravel* 12](#_Toc517457159)

[BAB III PERANCANGAN 13](#_Toc517457160)

[3.1 Perancangan Data 13](#_Toc517457161)

[3.2 Desain Sistem Secara Umum 13](#_Toc517457162)

[3.2.1 Perancangan Model *FP Tree* 15](#_Toc517457163)

[3.2.2. Perancangan Model *Classifier* 16](#_Toc517457164)

[3.2.3 Metode Evaluasi 17](#_Toc517457165)

[3.2.4 Perancangan Aplikasi 18](#_Toc517457166)

[BAB IV IMPLEMENTASI 23](#_Toc517457167)

[4.1 Lingkungan Implementasi 23](#_Toc517457168)

[4.2 Implementasi 23](#_Toc517457169)

[4.2.1 Implementasi FP Tree 23](#_Toc517457170)

[4.2.2 Implementasi *Classifier* 29](#_Toc517457171)

[4.2.3 Implementasi Evaluasi 32](#_Toc517457172)

[4.2.4 Implementasi Aplikasi 34](#_Toc517457173)

[BAB V PENGUJIAN DAN EVALUASI 53](#_Toc517457174)

[5.1 Lingkungan Pengujian 53](#_Toc517457175)

[5.2 Data Uji Coba 54](#_Toc517457176)

[5.3 Skenario Uji Coba 55](#_Toc517457177)

[5.4 Hasil Uji Coba 56](#_Toc517457178)

[5.4.1 Hasil Uji Sistem 56](#_Toc517457179)

[5.4.2 Hasil Uji Pengguna 61](#_Toc517457180)

[5.4.3 Hasil Uji Dokter 62](#_Toc517457181)

[5.5 Analisis Hasil Uji Coba 63](#_Toc517457182)

[BAB VI KESIMPULAN DAN SARAN 65](#_Toc517457183)

[6.1 Kesimpulan 65](#_Toc517457184)

[6.2 Saran 65](#_Toc517457185)

[DAFTAR PUSTAKA 67](#_Toc517457186)

[LAMPIRAN 69](#_Toc517457187)

[1. Hasil Lengkap Uji Sistem 69](#_Toc517457188)

[2. Hasil Lengkap Uji Dokter 77](#_Toc517457189)

[A. Uji Coba 1 77](#_Toc517457190)

[B. Uji Coba 2 78](#_Toc517457191)

[C. Uji Coba 3 78](#_Toc517457192)

[D. Uji Coba 4 79](#_Toc517457193)

[E. Uji Coba 5 80](#_Toc517457194)

[F. Uji Coba 6 81](#_Toc517457195)

[G. Uji Coba 7 82](#_Toc517457196)

[H. Uji Coba 8 83](#_Toc517457197)

[I. Uji Coba 9 84](#_Toc517457198)

[J. Uji Coba 6 85](#_Toc517457199)

***[Halaman ini sengaja dikosongkan]***

# DAFTAR GAMBAR

[Gambar 3.1 Diagram Sistem Secara Umum 14](#_Toc517457200)

[Gambar 3.2 Diagram Algoritma FP Tree 15](#_Toc517457201)

[Gambar 3.3 Diagram Alur Aplikasi 19](#_Toc517457202)

[Gambar 3.4 Diagram Algoritma Tanya Jawab 20](#_Toc517457203)

[Gambar 5.1 Grafik Hasil Uji Akurasi 57](#_Toc517457204)

[Gambar 5.2 Grafik Hasil Uji Retrieval 59](#_Toc517457205)

[Gambar 5.3 Grafik Hasil Uji Peringkat 60](#_Toc517457206)

***[Halaman ini sengaja dikosongkan]***

# DAFTAR TABEL

[Tabel 4.1 Lingkungan Implementasi Perangkat Lunak 23](#_Toc517457207)

[Tabel 5.1 Lingkungan Pengujian *Server* 53](#_Toc517457208)

[Tabel 5.2 Lingkungan Pengujian *Client* (Pengguna) 53](#_Toc517457209)

[Tabel 5.3 Jumlah Data Gejala dan Rules 54](#_Toc517457210)

[Tabel 5.4 Hasil Uji Akurasi 57](#_Toc517457211)

[Tabel 5.5 Hasil Uji Retrieval 58](#_Toc517457212)

[Tabel 5.6 Hasil Uji Peringkat 59](#_Toc517457213)

[Tabel 5.7 Hasil Uji Pengguna Dengan Pertanyaan Maksimal 30 61](#_Toc517457214)

[Tabel 5.8 Hasil Uji Pengguna Dengan Pertanyaan Maksimal 20 61](#_Toc517457215)

[Tabel 7.1 Hasil Pengujian *Naive Bayes*, *Min Support* = 5 69](#_Toc517457216)

***[Halaman ini sengaja dikosongkan]***

# DAFTAR KODE SUMBER

[Kode Sumber 4.1 Filter pada *FP Tree* 25](#_Toc517457236)

[Kode Sumber 4.2 Pembuatan Tree Pada FP Tree 26](#_Toc517457237)

[Kode Sumber 4.3 Ekstrak *Frequent Pattern* pada *FP Tree* 28](#_Toc517457238)

[Kode Sumber 4.4 Ekstrak Rules pada *FP Tree* 29](#_Toc517457239)

[Kode Sumber 4.5 Pembuatan Model *Naive Bayes* 31](#_Toc517457240)

[Kode Sumber 4.6 Pembuatan Model SVM 32](#_Toc517457241)

[Kode Sumber 4.7 Evaluasi Akurasi 32](#_Toc517457242)

[Kode Sumber 4.8 Evaluasi Information Retrieval 34](#_Toc517457243)

[Kode Sumber 4.9 Tanya Jawab Pada Server 41](#_Toc517457244)

[Kode Sumber 4.10 Prediksi Pada Server 44](#_Toc517457245)

[Kode Sumber 4.11 Interface HTML Pada User 47](#_Toc517457246)

[Kode Sumber 4.12 *Javascript* Tanya Jawab Pada *Interface User* 49](#_Toc517457247)

[Kode Sumber 4.13 *Javascript* Diagnosis Pada User 51](#_Toc517457248)

***[Halaman ini sengaja dikosongkan]***

# BAB I PENDAHULUAN

Bab ini akan membahas latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika laporan tugas akhir. Diharapkan dari penjelasan dalam bab ini gambaran tugas akhir secara umum dapat dipahami.

## Latar Belakang

Kemajuan teknologi membuat informasi semakin mudah diakses. Masyarakat kini lebih haus informasi dibanding dengan generasi sebelumnya. Tidak terkecuali dalam bidang kesehatan. Masyarakat kini dapat membaca artikel kesehatan dengan mudah. Mendapatkan obat dengan mudah. Obat obatan kini bukan lagi sebuah rahasia dokter dan apoteker. Masyarakat kini cenderung mengetahui jenis jenis obat dan apa guna obat tersebut.

Sakit dapat terjadi pada siapapun. Mulai dari penyakit ringan hingga penyakit serius. Menurut statistik dari *UK Digital Health Report*, 1 dari 5 orang lebih memilih untuk melakukan diagnosis sendiri dengan bantuan *search engine* [1]. Hal ini tentu saja meningkatkan resiko kesalahan diagnosis pada pasien, dan dapat menyebabkan penyakit pasien semakin memburuk bahkan meninggal dunia.

Untuk mengatasi hal tersebut, diperlukan suatu solusi yang dapat memperkecil kesalahan masyarakat dalam mendiagnosis penyakit. Sehingga pasien lebih waspada dan tidak menganggap remeh gejala yang mereka alami. Solusi tersebut dapat dikemas dalam bentuk aplikasi yang didukung oleh kecerdasan buatan yang dapat mendiagnosis berdasarkan gejala gejala yang diberikan oleh pasien.

Dalam tugas akhir ini, akan digunakan *FP Tree* untuk dapat memberikan gejala gejala berkaitan dengan gejala utama pasien dan dapat memberikan diagnosis yang sesuai.

## Rumusan Masalah

Rumusan masalah yang diangkat dalam tugas akhir ini dapat dipaparkan sebagai berikut :

1. Bagaimana memperoleh data gejala dan diagnosis yang sesuai untuk digunakan pada algoritma *Frequent Pattern Tree Growth*?
2. Bagaimana menentukan diagnosis yang cocok untuk gejala gejala yang telah diinput pasien berdasarkan hasil *Frequent Pattern Tree Growth*?
3. Bagaimana cara menghasilkan pertanyaan gejala yang sesuai dengan jawaban pertanyaan gejala sebelumnya menggunakan *Frequent Pattern Tree Growth*?

## Batasan Masalah

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan antara lain:

1. Jumlah diagnosis terbatas sebanyak 66 diagnosis.
2. Implementasi dilakukan pada lingkungan kerja berbasis web.

## Tujuan Pembuatan Tugas Akhir

Tujuan pembuatan tugas akhir ini adalah membangun aplikasi yang dapat memberikan diagnosis berdasarkan gejala gejala yang diberikan oleh pengguna menggunakan *Frequent Pattern Tree Growth*.

## Manfaat Tugas Akhir

Tugas akhir ini akan berguna untuk memberikan perkiraan penyakit kepada masyarakat awam, serta memberikan masukan diagnosis kepada dokter. Input data gejala dan kondisi pasien dapat dibaca oleh dokter yang ditunjuk oleh pasien sebagai pemeriksa dirinya. Sehingga akan membantu proses pemeriksaan dokter terhadap pasien.

## Metodologi

Metodologi yang akan digunakan pada tugas akhir ini adalah:

* 1. Penyusunan Proposal Tugas Akhir

Proposal akan berisi tentang pendahuluan, latar belakang, tujuan, manfaat, dan rumusan masalah. Yang akan didukung dengan penjelasan berupa tinjauan pustaka dan juga metode serta langkah langkah yang akan dilakukan untuk membangun produk.

* 1. Studi Literatur dan Wawancara

Sebelum pembuatan aplikasi, akan dilakukan studi literatur terkait aplikasi, dan juga melakukan wawancara kepada ahli, yang dalam tugas akhir ini adalah dokter untuk menyesuaikan metode yang dilakukan ahli untuk memberikan output yang diinginkan (diagnosis).

* 1. Analisis dan Desain

Pada tahap ini akan dilakukan analisis terkait bagaimana membangun model Frequent Pattern Tree Growth yang baik dan juga arsitektur terkait. Serta akan di analisa juga dataset yang sesuai untuk digunakan pada model.

* 1. Implementasi

Model yang telah dihasilkan pada langkah sebelumnya akan di implementasikan pada server yang berisi model, dan juga aplikasi untuk klien berupa web.

* 1. Evaluasi

Aplikasi yang telah dibuat akan dievaluasi oleh dokter dan juga pengguna untuk menilai kesesuaian anamnesis yang dilakukan oleh aplikasi dan kepuasan pengguna terhadap aplikasi. Selain itu aplikasi juga akan di evaluasi secara sistem menggunakan akurasi dengan rumus sebagai berikut

TP : True Positive TN : True Negative

FP : False Positive FN : False Negative

* 1. Penyusunan Buku Tugas Akhir

Pada tahap ini akan dilakukan penyusunan laporan berupa buku tugas akhir yang menjelaskan dasar teori pada tugas akhir ini serta hasil implementasi pada tugas akhir.

## Sistematika Penulisan Laporan Tugas Akhir

Buku tugas akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan tugas akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut ini.

1. Bab I. Pendahuluan

Bab ini berisi penjelasan mengenai latar belakang masalah, tujuan, dan manfaat dari pembuatan Tugas Akhir. Selain itu rumusan permasalahan, batasan masalah, dan sistematika penulisan juga merupakan bagian dari bab ini.

1. Bab II Tinjauan Pustaka

Bab ini berisi penjelasan tentang metode, algoritma,dan *library* yang digunakan pada pembuatan aplikasi ini. Pembahasan akan berfokus pada algoritma *associative rules* terutama pada *Frequent Pattern Tree Growth* dan juga metode klasifikasi yang digunakan yaitu *Naïve Bayes* dan *Support Vector Machine*.

1. Bab III Perancangan Perangkat Lunak

Bab ini aka membahas mengenai perancangan, desain, model, dan proses proses lain yang digunakan untuk pembuatan aplikasi juga model *Frequent Pattern Tree Growth* serta model *Classifier*.

1. Bab IV. Implementasi

Bab ini akan menjelaskan proses pembuatan aplikasi dengan bahasa *Python* yang dibantu oleh library *Flask* dan juga *PHP* dengan *framework laravel.* Juga akan membahas pembuatan model menggunakan bahasa *Python* yang didukung dengan *ScikitLearn*.

1. Bab V. Hasil Uji Coba dan Evaluasi

Bab ini akan menjelaskan hasil percobaan yang dilakukan secara sistem dengan *data testing* yang telah diambil dari sumber buku dan web, juga hasil percobaan yang dilakukan oleh dokter.

1. Bab VI. Kesimpulan dan Saran

Bab ini akan membahas kesimpulan yang didapatkan dari hasil ujicoba dan juga akan membahas saran saran yang didapatkan baik dari penulis juga dari dokter yang melakukan uji coba. Sehingga aplikasi dapat dikembangkan menjadi lebih baik

***[Halaman ini sengaja dikosongkan]***

# BAB II DASAR TEORI

## Anamnesis

Pada sub bab ini akan dijelaskan pengertian anamnesis dan bagaimana proses anamnesis yang dilakukan oleh dokter.

### 2.1.1 Pengertian Anamnesis

Proses akumulasi data yang menyangkut data medis pasien, latar belakang pasien, termasuk keluarga, lingkungan, pengalaman, terutama ingatan untuk digunakan dalam menganalisa kondisi [2].

### 2.1.2 Langkah Langkah Anamnesis

Umumnya anamnesis dilakukan sesuai dengan cara cara berikut [3]:

1. Pasien memberikan gejala

Pasien akan memberikan gejala yang dialami, misal : nyeri dada, badan panas.

1. Mendapatkan informasi lebih dalam mengenai gejala

Dokter akan menanyakan lebih lanjut terkait gejala yang dialami pasien. Misal pada nyeri dada, dokter akan menanyakan pertanyaan berikut kepada pasien.

* Dimana tepatnya letak nyeri pada dada?
* Sejak kapan nyeri dada dirasakan?
* Apakah nyeri dada sering sekali muncul, terkadang muncul, atau jarang muncul?
* Apakah rasa nyeri berpindah?
* Jika nyeri dada kambuh, berapa lama biasanya nyeri dada tersebut terasa?
* Apakah nyerinya semakin sakit atau semakin tidak sakit?
* Dalam skala 1-10, seberapa sakit yang anda rasakan?

1. Mencari gejala lain yang dialami pasien

Dokter akan menanyakan apakah pasien mengalami gejala lain yang mungkin berkaitan dengan gejala sebelumnya. Misal : Nyeri dada, maka mungkin pasien akan mengalami sulit bernapas.

1. Menanyakan tindakan/obat yang sudah dilakukan terhadap gejala tersebut

Dokter akan menanyakan tindakan atau obat apa yang telah dikonsumsi pasien. Lebih lanjut dokter akan menanyakan terkait dosis, nama obat, dan seberapa sering pasien mengonsumsi obat atau melakukan tindakan tersebut.

1. Menanyakan informasi kesehatan keluarga

Dokter akan menanyakan informasi keluarga yang mungkin memiliki penyakit yang berkaitan yang bersifat genetik. Misal : diabetes

1. Menanyakan informasi lingkungan keseharian

Dokter akan menanyakan bagaimana keseharian pasien, apakah pasien merokok, atau apakah pasien menggunakan obat obatan terlarang.

1. Menanyakan informasi lain terkait sistem tubuh lain yang tidak tercakup pada gejala

Dokter akan menanyakan apakah ada sistem tubuh lain yang terganggu. Umumnya sistem yang akan ditanyakan dokter adalah sebagai berikut :

* Cardiovaskular
* Pernapasan
* Pencernaan
* Saraf
* Genital
* Muskuloskeletal
* Kejiwaan

1. Mengulas ulang keluhan yang diberikan pasien

Dokter akan mengulas poin poin penting yang diberikan pasien sebelum memberikan diagnosis.

1. Dokter memberikan diagnosis

## 2.2 Frequent Pattern Tree Growth

Pada sub bab ini akan dijelaskan apa pengertian *Frequent Pattern Tree Growth,* istilah istilah pada algoritma tersebut dan juga algoritmanya.

### 2.2.1 Pengertian Frequent Pattern Tree Growth

Frequent Pattern Tree Growth atau FP Tree merupakan salah satu algoritma associative rules yang sering digunakan pada berbagai permasalahan data mining. Algoritma ini sendiri bertujuan membuat rules yang didasarkan pada tree yang dibuat berdasarkan dataset yang diberikan.

Seperti *tree* pada umumnya, *tree* pada FP Tree Growth juga memiliki root, node dan juga leaf. Pada FP Tree Growh penempatan node akan didasarkan pada support pada setiap attribute pada sebuah data. Sehingga jika dilihat semakin tinggi posisi dari suatu node maka dapat dipastikan node tersebut memiliki support yang lebih tinggi daripada *child*-nya.

### 2.2.2 Support

Support adalah batas minimal indikasi seberapa sering *item* akan muncul pada dataset. [4]

*support(A→B) = P (A∪ B)* (.)

**Keterangan** :

A dan B adalah atribut

### 2.2.3 Confidence

Confidence mengindikasikan jumlah dari pernyataan if-else yang bernilai benar.[4]

*confidence(A→B) = P(B|A)*  **(2.2)**

**Keterangan** :

A dan B adalah atribut

### 2.2.4 Algoritma FP Tree Growth

FP Tree Growth memiliki algoritma sebagai berikut [5]:

Input : Dataset, dan minimum support

Output : FP Tree

Langkah :

1. *Scan* *database*, dan mengumpulkan kumpulan *frequent items*, dan *minimum support* untuk setiap *frequent items*. Urutkan data tersebut sesuai dengan nilai *support* secara *descending*.
2. Buat *root* dari *tree*
3. Pilih salah satu *frequent item* dan buat *node* untuk setiap *item*. Lanjutkan hingga *item* dari *set* tersebut habis.
4. Jika *node* telah terbuat untuk *item* tertentu, maka atribut jumlah akan ditambahkan sesuai dengan frekuensi dia muncul pada *node* tersebut.
5. Ulangi langkah 3 dan 4 hingga *tree* terbuat.

## 2.4 Naïve Bayes

Naïve Bayes adalah salah satu algoritma *supervised* pada data mining. Naïve Bayes adalah sebuah *classifier* berbasis probabilitas yang sederhana yang menghitung frekuensi dan kombinasi nilai pada *dataset* [6]. Algoritma Naïve Bayes berbasiskan pada *Bayes Theorem*. Dengan persamaan sebagai berikut [7] :

(.)

**Keterangan**

A dan B adalah atribut

P(A) = Peluang A

P(B) = Peluang B

P (A | B) = Peluang A terhadap B

P (B | A) = Peluang B terhadap A

## 2.5 *Support Vector Machine*

*Support Vector Machine* atau SVM adalah sebuah *classifier* yang menggolongkan dengan cara membagi data menjadi area yang terpisah atau *hyperplane* [8]. Dengan adanya hyperplane tersebut data data yang berbeda dapat dikategorikan menjadi 2 data berbeda. *Hyperplane* terbaik adalah *hyperplane* yang dapat memberikan margin terbesar dari 2 kategori [9].

## 2.6 *Scikit Learn*

*Scikit Learn* adalah library machine learning gratis berbasiskan bahasa pemrograman *Python*. Library ini memiliki banyak fitur mulai dari klasifikasi, regresi, kluster termasuk *SVM, Random Forest, Gradient Boosting, KMeans, DBScan* dan didesain untuk bekerja dengan *library* numerik dan ilmiah milik Python, *NumPy* dan *SciPy* [10].

## 2.7 *Python – Flask*

*Flask* adalah sebuah *microframework* untuk python [11]. *Flask* umumnya digunakan untuk membuat web. Flask tidak memiliki *database abstraction layer, form validation,* dan banyak fitur lainnya yang mana *library* lain sudah ada dan dapat mengatasi masalah tersebut. *Flask* mungkin *library* mikro namun mampu untuk digunakan dalam berbagai kebutuhan [12].

## 2.8 *PHP - Laravel*

*Laravel* adalah sebuah *framework* untuk *web* dengan *syntax* yang ekspresif dan elegan. *Laravel* mudah diakses, kuat, dan memiliki *tools* untuk aplikasi besar dan kuat. [13]. Berbeda dengan *flask*, *laravel* berbasiskan bahasa pemrograman PHP.

# BAB III PERANCANGAN

## 3.1 Perancangan Data

Data yang akan digunakan pada aplikasi ini berasal dari berbagai sumber buku dan *website*. Penulis mengekstrak informasi data dari buku dan *website* dan memasukkannya pada *database*.

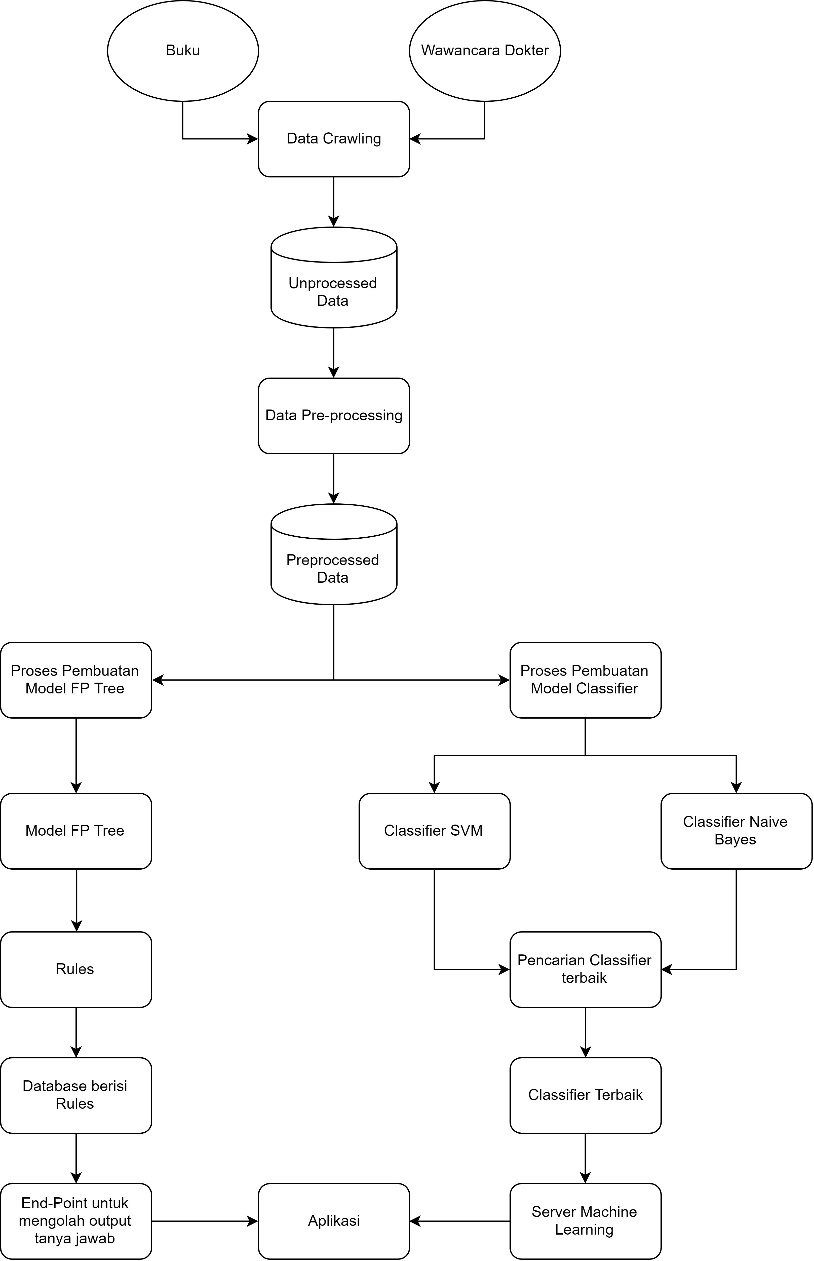
Data ini terdiri dari 858 data kasus, yang memiliki 66 diagnosis atau kelas. Serta 375 atribut. Data ini akan dibagi menjadi 2 kelompok yaitu data belajar dan data pengujian. Data belajar terdiri dari 10 kasus untuk setiap kelas. Dan data pengujian terdiri dari 3 kasus untuk setiap kelas.

## 3.2 Desain Sistem Secara Umum

Rancangan aplikasi terbagi dalam 3 tahap. Tahap pembuatan model FP Tree, tahap pembuatan model *classifier*, dan tahap pembuatan aplikasi untuk pengguna. Pada tahap pembuatan model *FP Tree*, data akan difilter dengan *minimum support* sesuai dengan skenario. Lalu data yang telah di *filter* sesuai *minimum support* tersebut akan di training untuk membuat *rules*. *Rules* yang dihasilkan oleh *FP Tree* akan dimasukkan kedalam *database* yang nantinya akan digunakan oleh *server* untuk menentukan pertanyaan yang akan ditanyakan kepada pengguna.

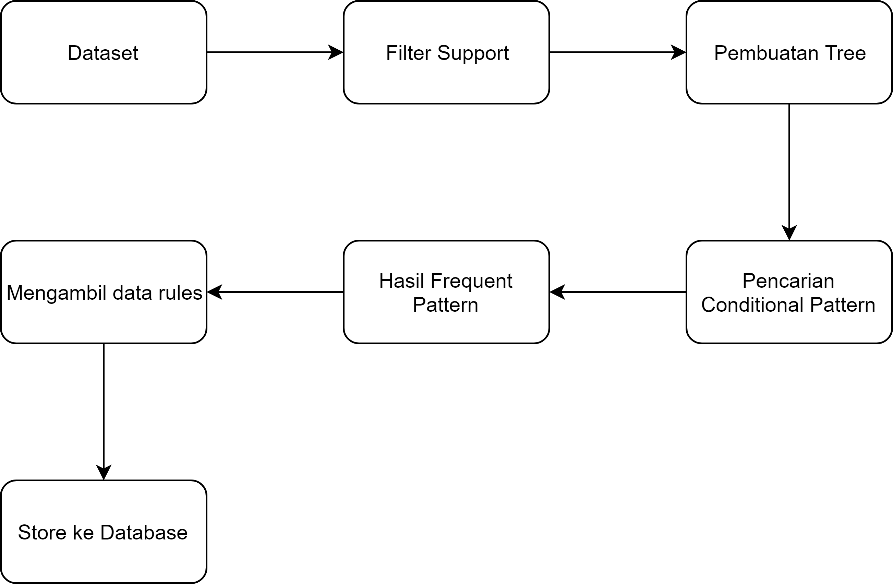
Pada tahap pembuatan model *classifier,* dataset akan di filter berdasarkan *minimum support*, yang mana akan di *training* dengan *Naïve Bayes* dan juga *SVM.* Lalu masing masing hasil model akan di ujikan dengan data pengujian. Hasil dari pengujian ini adalah *classifer* dengan hasil yang terbaik, yang mana model ini yang akan digunakan untuk server *machine learning* yang akan menentukan diagnosis penyakit.

Tahap pembuatan aplikasi user akan menggabungkan antara hasil rules dengan model *classifier*. Aplikasi ini akan dibuat dengan backend *Laravel* dan *front-end Javascript Vanilla*.



Gambar . Diagram Sistem Secara Umum

### 3.2.1 Perancangan Model *FP Tree*



Gambar . Diagram Algoritma FP Tree

*Dataset* akan diolah dengan algoritma *FP Tree* sesuai dengan algoritma pada bab sebelumnya. Algoritma akan diimplementasikan dengan bahasa pemrograman *Python*.

Pada tahap *filter* atribut pada masing masing data akan di *filter* berdasarkan *minimum support*. Adapun *minimum support* yang akan diskenariokan akan dijelaskan pada bab berikutnya.

Setelah di *filter*, data akan diurutkan berdasarkan urutan *support* dari yang terbesar hingga terkecil. Dan urutan tersebut akan menunjuk urutan sebelumnya yang lebih besar sehingga atribut yang memiliki *support* terbesar akan terletak pada *layer* teratas dan *support* terkecil akan terletak pada *layer* terbawah. Jika ada data yang memiliki atribut sama pada sebuah *layer* yang sama dengan *parent* yang sama maka *tree* tidak perlu membuat *node* baru, melainkan hanya menambahkan atribut frekuensi pada node yang sama tersebut.

Setelah tree terbentuk, maka akan dicari *Conditional Pattern* dari setiap *node*. Pencarian ini akan mencari *path* dari *node* tersebut hingga ke *root*. Dari proses ini akan memunculkan *paths* yang dilalui oleh *node* yang memiliki atribut yang sama.

Dari *Conditional Pattern,* kita dapat mengekstrak *Frequent Pattern.* *Frequent Pattern* diambil dari kombinasi *Conditional Pattern* yang memiliki support lebih dari *minimum support*.

Dari hasil *Frequent Pattern*, akan diekstrak kombinasi yang akan memunculkan *rules*. *Rules* tersebut akan dihitung *confidence* dengan cara membagi hasil kombinasi dengan *pattern* utama. *Confidence* tidak akan di *filter* mengingat jumlah *rules* yang tidak begitu banyak.

*Rules* tersebut akan disimpan pada sebuah tabel pada *database* yang memiliki atribut *id, rules input, rules output* dan *probability*. Sehingga nantinya pada akhir proses akan memunculkan 9 tabel sesuai dengan jumlah skenario.

### 3.2.2. Perancangan Model *Classifier*

Pada sub bab ini akan dijelaskan bagaimana perancangan model *classifier* yaitu *Naïve Bayes dan* juga *SVM*.

#### 3.2.2.1 *Naïve Bayes*

*Naïve Bayes* yang digunakan pada model adalah *Gaussian Naïve Bayes* karena dataset memiliki *format* *binary*. Model *Naïve Bayes* akan dibuat berdasarkan *dataset* yang telah di filter berdasarkan *minimum support* yang sama dengan skenario. Setelah itu *dataset* yang telah di filter tersebut akan di masukkan sebagai data belajar dari model *Naïve Bayes* yang akan dibuat. Model *Naïve Bayes* akan dibuat menggunakan *library* *2.6 Scikit* Learn. Nantinya model ini akan diujikan dengan data *testing* yang telah ditetapkan sebelumnya.

*Output* dari model ini adalah 5 kemungkinan diagnosis yang memiliki *probability* tertinggi. Dengan format diagnosis dan persentase kemungkinan untuk setiap diagnosis.

#### 3.2.2.2 Support Vector Machine

Pada tahap ini akan dibuat model SVM dengan *kernel linear*. Pada *Scikit Learn* lebih dikenal dengan SVC. Sama seperti *Naïve Bayes*, model SVM juga akan dibuat berdasarkan data belajar yang telah di filter berdasarkan *minimum support*. Model ini dibuat dengan library *Scikit Learn*.

*Output* dari model ini adalah 5 kemungkinan diagnosis yang memiliki *probability* tertinggi. Dengan format diagnosis dan persentase kemungkinan untuk setiap diagnosis.

### 3.2.3 Metode Evaluasi

Pada sub bab ini akan dijelaskan metode evaluasi dari tugas akhir ini. Evaluasi ini nantinya akan dibagi menjadi 2 yaitu berdasarkan akurasi dan juga *information retrieval*.

#### 3.2.3.1 Evaluasi Berdasarkan Akurasi

Metrik akurasi akan didasarkan pada hasil klasifikasi dengan probabilitas klasifikasi tertinggi dari model *classifier* dengan hasil yang seharusnya. Dengan rumus sebagai berikut :

(3.1)

**Keterangan** :

FP = *False Positive*

FN = *False Negative*

TP = *True Positive*

TN = *True Negative*

#### 3.2.3.2 Evaluasi Berdasarkan Information Retrieval

Metrik ini dihitung berdasarkan hasil pada data training yang akan dibandingkan dengan hasil model yang memiliki 5 hasil probabilitas tertinggi. Dan data benar tersebut akan dihitung berdasarkan peringkat, 100 untuk peringkat pertama dan turun 20 untuk setiap peringkat sehingga peringkat 5 akan mendapatkan nilai 20 dan jika data tidak muncul akan memberikan nilai 0. Atau disederhanakan menjadi rumus berikut :

(.)

**Keterangan** :

Peringkat merupakan urutan diagnosis berdasarkan probabilitas

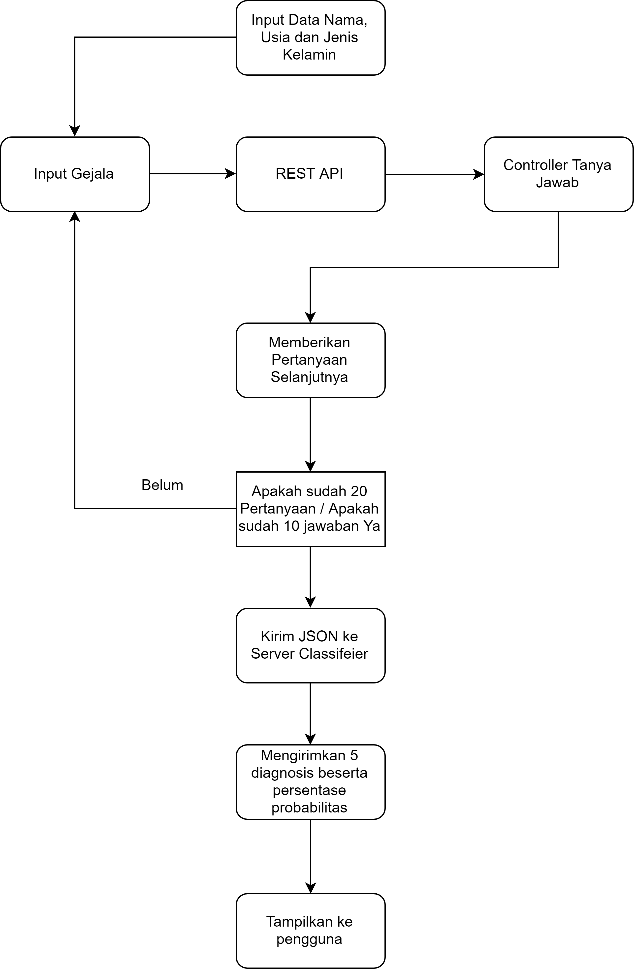
Selain itu juga akan dihitung nilai kemunculan dari hasil diagnosis. Sehingga jika ada salah satu hasil klasifikasi dari 5 hasil yang sesuai dengan hasil pada data training maka akan dinilai benar.

### 3.2.4 Perancangan Aplikasi

Pada sub-bab ini akan dijelaskan bagaimana perancangan desain aplikasi. Perancangan aplikasi akan dibagi menjadi 3 yaitu :

* 1. Perancangan Desain Aplikasi Secara Umm
  2. Perancangan Output Tanya Jawab
  3. Perancangan Output Diagnosis

#### 3.2.4.1 Perancangan Desain Aplikasi Secara Umum

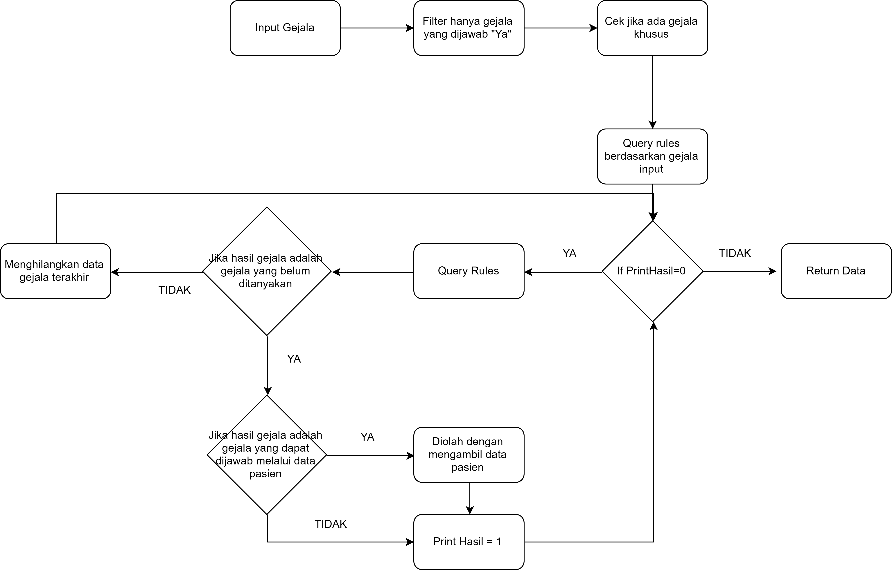


Gambar . Diagram Alur Aplikasi

Aplikasi akan dimulai dengan menanyakan identitas pasien, nama, usia dan juga jenis kelamin. Setelah itu pasien akan menginputkan salah satu gejala yang dirasa paling mengganggu pasien. Setelah itu aplikasi akan menanyakan pertanyaan gejala yang mungkin berikutnya berdasarkan jawaban dari pertanyaan gejala sebelumnya. Pertanyaan akan memiliki 2 jawaban ya atau tidak.

Aplikasi akan menghentikan pertanyaan dan akan memberikan jawaban jika aplikasi telah menanyakan 20 aplikasi atau pasien telah memberikan 10 jawaban “ya” pada pertanyaan yang ditanyakan.

#### 3.2.4.2 Perancangan *Output* Tanya Jawab



Gambar . Diagram Algoritma Tanya Jawab

Agar aplikasi dapat menentukan pertanyaan mana yang harus ditanyakan selanjutnya, aplikasi akan mengakses *end-point* yang bertugas untuk menentukan pertanyaan selanjutnya. Adapun *end-point* ini akan menentukan pertanyaan selanjutnya berdasarkan jawaban pertanyaan sebelumnya. Jawaban tersebut akan diolah oleh *server* dengan cara mengambil data *rules* yang berasal dari output model *FP Tree*, dan akan menentukan kemungkinan pertanyaan selanjutnya berdasarkan input berupa gejala sebelumnya.

Adapun olahan khusus, seperti gejala yang memiliki sub gejala, seperti demam yang memiliki sub “demam tinggi” dan “demam lebih dari 3 hari”, maka mesin akan langsung menanyakan sub gejala sub gejala tersebut jika telah mendapai salah satu sub gejala ditanyakan. Serta pengolahan data khusus yang didasari dari input identitas pasien yaitu usia dan jenis kelamin. Adapun berikut adalah pembagian kategori menurut usia :

1. Bayi 0 – 1
2. Balita dan Remaja 3 - 18
3. Dewasa 18 – 65
4. Tua > 65

#### 3.2.4.3 Perancangan *Output* Diagnosis

Jika aplikasi sudah selesai menanyakan pertanyaan, maka aplikasi akan mengirimkan data gejala yang dialami pasien ke server *machine learning* yang dijalankan menggunakan *flask* dan akan mengolahnya pada model *machine learning*. Setelah itu *server* ini akan mengirimkan kembali hasil data ke aplikasi. Dan aplikasi akan menampilkan 5 kemungkinan diagnosis tertinggi beserta persentase.

***[Halaman ini sengaja dikosongkan]***

# BAB IV IMPLEMENTASI

Pada bab ini akan dijelaskan mengenai implementasi yang dilakukan berdasarkan rancangan yang telah dijelaskan pada bab sebelumnya. Implementasi kode program bahasa yang berbeda beda yang akan dijelaskan setiap potongan kode.

## Lingkungan Implementasi

Spesifikasi komputer (*server*) yang digunakan untuk implementasi aplikasi ditampilkan pada Tabel 4.1

Tabel . Lingkungan Implementasi Perangkat Lunak

|  |  |
| --- | --- |
| **Komponen** | **Spesifikasi** |
| CPU | Intel® Core i5™ 7200U CPU @ 2.50 GHz (4 CPUs), ~ 2.7GHz |
| GPU | NVidia GeForce 720 930MX 2GB |
| Sistem Operasi | Windows 10 Pro |
| Memori | 8GB DDR4 |
| Penyimpanan | 1 TB |
| Perangkat Lunak Pendukung | * Python 3.6 * Python 2.7 * PHP 7.1 * MySQL 5.5 |

## Implementasi

Sub-bab ini akan menjelaskan implementasi dari setiap tahap tahap yang telah dijelaskan pada bab sebelumnya.

### Implementasi FP Tree

Pada sub bab ini akan dijelaskan tahap tahap pembuatan model *FP Tree* yang akan menghasilkan rules. Tahap pembuatan FP Tree ini sendiri ada 3 tahap yaitu :

* 1. *Filter Support*
  2. Pembuatan *Tree*
  3. Ekstrak *Frequent Pattern*
  4. Ekstrak *Rules*

Pada tahap ini semua implementasi kode sumber menggunakan bahasa *Python*.

#### *Filtering Support*

Pada tahap ini data akan diolah oleh suatu fungsi yang akan memfilter data sesuai dengan *minimum support* yang dimiliki data tersebut. Pada baris ke 5 – 7 atribut pada gejala akan di *filter* sesuai *minimum support*. Dan di baris ke 11 – 12 data akan diurutkan. Dan pada baris ke 15 – 19 data akan di filter sesuai dengan urutan dan atribut mana yang melebihi *minimum support*.

1. **def** cleanData**(**data**,**min\_support**):**
2. items **=** data
3. item\_counter **=** Counter**(**x **for** sublist **in** data **for** x **in** sublist**)**
4. temp\_item\_counter **=** Counter**(**item\_counter**)**
5. **for** item **in** temp\_item\_counter**:**
6. **if(**temp\_item\_counter**[**item**]** **<** min\_support**):**
7. **del** item\_counter**[**item**]**
8. itemOrder **=** **{}**
9. lengthItems **=** len**(**item\_counter**)**
10. **for** i **in** range**(**1**,**lengthItems**+**1**):**
11. itemOrder**[**item\_counter**.**most\_common**()[-**i**][**0**]]** **=** i**;**
12. count **=** 0**;**
13. **for** item **in** items**:**
14. item **=** filter**(lambda** v**:** v **in** itemOrder**,** item**)**
15. item**.**sort**(**key**=lambda** v**:** itemOrder**[**v**],** reverse**=True)**
16. items**[**count**]** **=** item
17. count**+=**1
18. data **=** **[**items**,**item\_counter**]**
19. **return** data

Kode Sumber . Filter pada *FP Tree*

#### Pembuatan *Tree*

Pada tahap ini akan data yang telah di filter akan dimasukkan ke sebuah *tree*. Output dari fungsi ini adalah sebuah *tree*, yang mana setiap *node* nya memiliki atribut nama, *child*, *parent*, dan juga *support*.

1. **def** buildTree**(**dataset**):**
2. root **=** Node**(**"root"**,**support **=** 0**)**
3. **for** data **in** dataset**:**
4. parent **=** 0**;**
5. **for** item **in** data**:**
6. **if** parent**==**0**:**
7. parent **=** root
8. child **=** find**(**parent**,** filter\_**=lambda** node**:** node**.**name **==** item**,** stop**=None,** maxlevel **=** 2**)**
9. **if** child**:**
10. child**.**support **=** child**.**support **+** 1
11. parent **=** child
12. **else:**
13. new\_node **=** Node**(**item**,** parent**=**parent**,** support **=** 1**)**
14. parent **=** new\_node
15. **return** root

Kode Sumber . Pembuatan Tree Pada FP Tree

#### Mengekstrak Frequent Pattern

Pada tahap ini dari *tree* fungsi akan mengekstrak *frequent patt*ern yang digunakan untuk membuat *rules* pada tahap berikutnya. Pada awal fungsi yaitu baris 3 – 6 *tree* akan di *reverse* dari bawah. Selanjutnya *node node* pada *tree* akan di jelajahi satu per satu untuk mencari *frequent pattern* terbanyak.

1. **def** mineFrequentPattern**(**dataset**,** tree**,** min\_support**,**itemCounter**):**
2. root **=** tree
3. itemReverse **=** **[]**
4. lengthItems **=** len**(**itemCounter**)**
5. **for** i **in** range**(**1**,**lengthItems**+**1**):**
6. itemReverse**.**append**(**itemCounter**.**most\_common**()[-**i**][**0**])**
7. condPattern **=** **{}**
8. frequentPattern **=** **[]**
9. **for** item **in** itemReverse**:**
10. nodes **=** findall\_by\_attr**(**root**,** item**)**
11. condPatternParents **=** **[]**
12. **for** node **in** nodes**:**
13. parent **=** str**(**node**)**
14. parent **=** parent**.**replace**(**"/root/"**,**""**)**
15. parent **=** parent**.**split**(**"'"**)[**1**]**
16. **for** i **in** range**(**0**,** node**.**support**):**
17. condPatternParents**.**append**(**parent**)**
18. condPattern**[**item**]** **=** condPatternParents
19. **for** item **in** itemReverse**:**
20. itemsInPattern **=** **[]**
21. **for** pattern **in** condPattern**[**item**]:**
22. temp\_pattern **=** pattern**.**split**(**'/'**)**
23. **for** temp\_item **in** temp\_pattern**:**
24. itemsInPattern**.**append**(**temp\_item**)**
25. item\_counter **=** Counter**(**itemsInPattern**)**
26. temp\_item\_counter **=** Counter**(**item\_counter**)**
27. **for** temp\_item **in** temp\_item\_counter**:**
28. **if(**temp\_item\_counter**[**temp\_item**]** **<** min\_support**):**
29. **del** item\_counter**[**temp\_item**]**
30. willBeCombinatedItem **=** **[]**
31. willBeCombinatedItem **=** filter**(lambda** v**:** v **in** item\_counter**,** itemReverse**)**
32. **for** L **in** range**(**1**,** len**(**willBeCombinatedItem**)+**1**):**
33. **for** resultcombo **in** combinations**(**willBeCombinatedItem**,** L**):**
34. resultcombo **=** list**(**resultcombo**)**
35. **if** item **in** resultcombo**:**
36. support **=** 0
37. **for** pattern **in** condPattern**[**item**]:**
38. pattern\_array **=** pattern**.**split**(**'/'**)**
39. flag\_support\_increment **=** 1
40. **for** combo **in** resultcombo**:**
41. **if** combo **not** **in** pattern\_array**:**
42. flag\_support\_increment **=** 0
43. **if** flag\_support\_increment **==** 1**:**
44. support **+=** 1
45. **if** support **>=** min\_support**:**
46. frequentPattern**.**append**(**FrequentPattern**(**item**,**','**.**join**(**resultcombo**) ,**str**(**support**)))**
47. **return** frequentPattern

Kode Sumber . Ekstrak *Frequent Pattern* pada *FP Tree*

#### Ekstrak *Rules*

Setelah berhasil mengekstrak *frequent pattern*, program selanjutnya akan mengekstrak *rules* dari data *frequent pattern* dan akan dimasukkan kedalam database.

1. **def** generate\_rules**(**filteredList**,**threshold**):**
2. **for** i **in** range**(**len**(**filteredList**)):**
3. subset **=** filteredList**[**i**].**pattern**.**split**(**','**);**
4. **for** L **in** range**(**1**,** len**(**subset**)):**
5. **for** resultsubset **in** combinations**(**subset**,** L**):**
6. left **=** list**(**resultsubset**)**
7. right **=** list**(**subset**)**
8. **for** item **in** left**:**
9. right**.**remove**(**item**)**
10. left\_string **=** ','**.**join**(**left**)**
11. right\_string **=** ','**.**join**(**right**)**
12. subset\_string **=** ','**.**join**(**subset**)**
13. support\_subset **=** find\_by\_item**(**filteredList**,** subset\_string**)**
14. support\_left **=** find\_by\_item**(**filteredList**,** left\_string**)**
15. support **=** float**(**float**(**support\_subset**)/**float**(**support\_left**))**
16. right **=** sorted**(**right**,** key**=lambda** x**:** int**(**x**))**
17. left **=** sorted**(**left**,** key**=lambda** x**:** int**(**x**))**
18. left\_string **=** ','**.**join**(**left**)**
19. right\_string **=** ','**.**join**(**right**)**
20. left\_string **=** left\_string**.**replace**(**" "**,**""**)**
21. right\_string **=** right\_string**.**replace**(**" "**,**""**)**
22. query **=** 'INSERT INTO rules\_23(items,result,probability) VALUES("'**+**left\_string**+**'","'**+**right\_string**+**'","'**+**str**(**support**)+**'");'
23. cursor**.**execute**(**query**)**
24. cnx**.**commit**()**

Kode Sumber . Ekstrak Rules pada *FP Tree*

### Implementasi *Classifier*

Pada sub bab ini akan dijelaskan bagaimana tahapan pembuatan model *classifier*. Adapun pada tugas akhir ini *classifier* yang digunakan ada 2 yaitu :

* 1. *Naïve B*ayes
  2. *Support Vector Machine*

Semua implementasi pada kode sumber pada tahap ini menggunakan bahasa *Python*.

#### *Naïve Bayes*

Berikut merupakan implementasi kode dari pembuatan model *Naïve Bayes*, yang mana *output-*nya adalah berupa *file* model yang dapat di buka lagi tanpa perlu *training* oleh *server* *machine learning*.

1. **def** generate\_model**():**
2. seed **=** 7
3. numpy**.**random**.**seed**(**seed**)**
4. dataframe **=** pandas**.**read\_csv**(**"../Users/Kevin/PycharmProjects/TugasAkhir/dataset/classifier-training-t5.csv"**,** skipinitialspace**=True)**
5. dataset **=** dataframe**.**values
6. jumlah\_gejala **=** len**(**dataset**[**0**])** **-** 1
7. X\_train **=** dataset**[:,**0**:**jumlah\_gejala**]**
8. Y\_train **=** dataset**[:,**jumlah\_gejala**]**
9. dataframe **=** pandas**.**read\_csv**(**"../Users/Kevin/PycharmProjects/TugasAkhir/dataset/classifier-testing-t5.csv"**,** skipinitialspace**=True)**
10. dataset **=** dataframe**.**values
11. X\_test **=** dataset**[:,**0**:**jumlah\_gejala**]**
12. Y\_test **=** dataset**[:,**jumlah\_gejala**]**
13. label **=** pandas**.**read\_csv**(**"../Users/Kevin/PycharmProjects/TugasAkhir/dataset/label.csv"**)**
14. label **=** label**.**values
15. model **=** BernoulliNB**()**
16. y\_pred **=** model**.**fit**(**X\_train**,** Y\_train**).**predict**(**X\_test**)**
17. class\_map **=** model**.**classes\_
18. score **=** model**.**score**(**X\_test**,** Y\_test**)**
19. filename **=** '../Users/Kevin/PycharmProjects/TugasAkhir/NaiveBayes/model\_architecture.sav'
20. pickle**.**dump**(**model**,** open**(**filename**,** 'wb'**))**

Kode Sumber . Pembuatan Model *Naive Bayes*

#### Support Vector Machine

Berikut merupakan implementasi kode dari pembuatan model Support Vector Machine, yang mana *output-*nya adalah berupa *file* model yang dapat di buka lagi tanpa perlu *training* oleh *server machine learning*.

1. **def** generate\_model**():**
2. seed **=** 7
3. numpy**.**random**.**seed**(**seed**)**
4. dataframe **=** pandas**.**read\_csv**(**"../Users/Kevin/PycharmProjects/TugasAkhir/dataset/classifier-training-t5.csv"**,** skipinitialspace**=True)**
5. dataset **=** dataframe**.**values
6. jumlah\_gejala **=** len**(**dataset**[**0**])** **-** 1
7. X\_train **=** dataset**[:,**0**:**jumlah\_gejala**]**
8. Y\_train **=** dataset**[:,**jumlah\_gejala**]**
9. dataframe **=** pandas**.**read\_csv**(**"../Users/Kevin/PycharmProjects/TugasAkhir/dataset/classifier-testing-t5.csv"**,** skipinitialspace**=True)**
10. dataset **=** dataframe**.**values
11. X\_test **=** dataset**[:,**0**:**jumlah\_gejala**]**
12. Y\_test **=** dataset**[:,**jumlah\_gejala**]**
13. label **=** pandas**.**read\_csv**(**"../Users/Kevin/PycharmProjects/TugasAkhir/dataset/label.csv"**)**
14. label **=** label**.**values
15. model = SVC(kernel = 'linear', probability=True).fit(X\_train, Y\_train)y\_pred **=** model**.**fit**(**X\_train**,** Y\_train**).**predict**(**X\_test**)**
16. class\_map **=** model**.**classes\_
17. score **=** model**.**score**(**X\_test**,** Y\_test**)**
18. filename **=** '../Users/Kevin/PycharmProjects/TugasAkhir/NaiveBayes/model\_architecture.sav'
19. pickle**.**dump**(**model**,** open**(**filename**,** 'wb'**))**

Kode Sumber . Pembuatan Model SVM

### Implementasi Evaluasi

Pada tahap ini akan dijelaskan bagaimana *classifier* melakukan evaluasi dengan masing masing model. Bahasa yang digunakan untuk implementasi pada sumber kode semua dalam bahasa *Python*.

#### Implementasi Evaluasi Akurasi

Berikut adalah implementasi evaluasi akurasi, yang mana akan mengukur akurasi dari model berdasarkan input dan output data *testing*.

1. **def** getScore**():**
2. filename = '../Users/Kevin/PycharmProjects/TugasAkhir/NaiveBayes/model\_architecture.sav'
3. loaded\_model = pickle.load(open(filename, 'rb'))
4. score = loaded\_model.score(X\_test, Y\_test)
5. return score

Kode Sumber . Evaluasi Akurasi

#### Implementasi Evaluasi Retrieval

Pada tahap ini ada 2 penilaian yang dilakukan. Yaitu penilaian berdasarkan kemunculan diagnosis benar dari 5 diagnosis yang di prediksi dan juga penilaian peringkat kemunculan diagnosis benar dari 5 diagnosis.

1. **def** getIRScore**():**
2. predictions **=** model**.**predict\_proba**(**X\_test**)**
3. index **=** 0
4. filewrite **=** '../Users/Kevin/PycharmProjects/TugasAkhir/laporan/t18/svm\_prediction\_result.txt'
5. **with** open**(**filewrite**,** 'w'**)** **as** result\_file**:**
6. result\_file**.**write**(**'jumlah gejala ' **+** str**(**jumlah\_gejala**)** **+** '\n'**)**
7. result\_file**.**write**(**'jumlah diagnosis ' **+** str**(**len**(**class\_map**))** **+** '\n'**)**
8. result\_file**.**write**(**'akurasi ' **+** str**(**score**)** **+** '\n\n'**)**
9. item\_pred\_IR **=** 0**;**
10. pred\_class\_score\_array **=** **[]**
11. item\_count **=** 0**;**
12. **for** pred **in** predictions**:**
13. top5 **=** pred**.**argsort**()[-**5**:][::-**1**]**
14. item\_pred **=** top5**[**0**]**
15. item\_pred\_str **=** class\_map**[**item\_pred**]**
16. item\_true **=** Y\_test**[**index**]**
17. string **=** str**(**item\_pred\_str**)** **+** ' --- ' **+** str**(**item\_true**)**
18. result\_file**.**write**(**string**+**'\n'**)**
19. pred\_class\_score **=** 100**;**
20. pred\_class\_score\_fix **=** 0**;**
21. **for** item **in** top5**:**
22. result\_file**.**write**(**str**(**class\_map**[**item**])** **+** ' '**+** str**(**pred**[**item**])+**'\n'**)**
23. temp\_pred **=** str**(**class\_map**[**item**])**
24. temp\_true **=** str**(**item\_true**)**
26. **if(**pred\_class\_score\_fix **==** 0**):**
27. **if(**temp\_true **==** temp\_pred**):**
28. pred\_class\_score\_fix **=** pred\_class\_score
29. item\_pred\_IR **+=** 1
30. **else:**
31. pred\_class\_score **=** pred\_class\_score **-** 20**;**
32. pred\_class\_score\_array**.**append**(**pred\_class\_score\_fix**)**
33. result\_file**.**write**(**'skor IR' **+** str**(**pred\_class\_score\_fix**)** **+** '\n'**)**
34. result\_file**.**write**(**'\n'**)**

Kode Sumber . Evaluasi Information Retrieval

### Implementasi Aplikasi

Implementasi terakhir adalah implementasi pada aplikasi. Semua output dari tahap tahap sebelumnya akan disatukan sehingga dapat digunakan oleh user. Adapun pada tahap pembuatan aplikasi ini dibagi menjadi 3 bagian yaitu :

* + 1. Tanya Jawab
    2. Prediksi
    3. *Interface*

#### Tanya Jawab

Pada bagian ini, *server* akan menerima *input* hasil dari pertanyaan terbaru dan pertanyaan sebelumnya. Lalu *server* akan melakukan pengolahan data. Dan mengambil *rules* dari *database* dan akan memberikan pertanyaan berikutnya berdasarkan probabilitas terbesar. Implementasi dari bagian ini menggunakan bahasa PHP.

1. public function getNextQuestion(Request $request)
2. {
3. $gejala = $request->gejala;
4. $pasien['jenis\_kelamin'] = $request->jenis\_kelamin;
5. $pasien['usia'] = $request->usia;
6. $arrayGejala = [];
7. $arrayGejalaAsked = [];
8. $arrayGejalaSpecial = [49,35,48,331];
9. $arrayGejalaSpecialUsia = [139,35,48,331];
10. foreach($gejala as $key=>$value) {
11. if($value == 1)
12. array\_push($arrayGejala, $key);
13. array\_push($arrayGejalaAsked, $key);
14. }
16. if(in\_array(10, $arrayGejalaAsked) && $gejala['10'] == 1)
17. {
18. $id = 0;
19. if(!in\_array(57, $arrayGejalaAsked) && !in\_array(66, $arrayGejalaAsked))
20. {
21. $id = 57;
22. }
23. elseif(in\_array(57, $arrayGejalaAsked) && $gejala['57'] == 0 && !in\_array(66, $arrayGejalaAsked))
24. {
25. $id = 66;
26. }
27. elseif(in\_array(66, $arrayGejalaAsked) && $gejala['66'] == 0 && !in\_array(57, $arrayGejalaAsked))
28. {
29. $id = 57;
30. }
31. if($id!=0)
32. {
33. $rules = Gejala::where('id',$id)->first();
34. return json\_encode([
35. 'command' => 'ask',
36. 'gejala' => $rules,
37. 'append' => 0,
38. 'append\_value' => 0
39. ]);
40. }
41. }
43. if(in\_array(23, $arrayGejalaAsked) && $gejala['23'] == 1)
44. {
45. $id = 0;
46. if(!in\_array(41, $arrayGejalaAsked))
47. {
48. $id = 41;
49. }
50. elseif(!in\_array(177, $arrayGejalaAsked))
51. {
52. $id = 177;
53. }
54. if($id!=0)
55. {
56. $rules = Gejala::where('id',$id)->first();
57. return json\_encode([
58. 'command' => 'ask',
59. 'gejala' => $rules,
60. 'append' => 0,
61. 'append\_value' => 0
62. ]);
63. }
64. }
65. foreach($arrayGejalaSpecialUsia as $age)
66. {
67. if(in\_array($age, $arrayGejalaAsked))
68. {
69. array\_push($arrayGejalaAsked, 35);
70. array\_push($arrayGejalaAsked, 48);
71. array\_push($arrayGejalaAsked, 331);
72. array\_push($arrayGejalaAsked, 139);
73. break;
74. }
75. }
76. $arrayGejalaSorted = $arrayGejala;
77. $tempArrayGejala = $arrayGejala;
78. sort($arrayGejalaSorted);
79. $stringGejalaSorted = implode(",", $arrayGejalaSorted);
80. $printResult = 0;
81. $gejala\_data = 0;
82. $append = 0;
83. $append\_value = 0;
84. $command = 'ask';
85. $out = 0;
86. $found = 0;
87. $while\_iterate\_count = 0;
88. while(!$printResult)
89. {
90. $rules = Rules::where('items',$stringGejalaSorted)->orderBy('probability','desc')->get();
91. if(!empty($rules[0]))
92. {
93. if($while\_iterate\_count > 20)
94. {
95. $printResult = 1;
96. $command = 'out';
97. }
98. foreach($rules as $rule)
99. {
100. if(!in\_array($rule->result, $arrayGejalaAsked))
101. {
102. if(in\_array($rule->result, $arrayGejalaSpecial))
103. {
104. $gejala\_result = explode(',', $rule->result);
105. $result = $this->getSpecialRule($gejala\_result[0],$pasien);
106. $command = $result['command'];
107. $gejala\_data = $result['gejala'];
108. $append = $result['append'];
109. $append\_value = $result['append\_value'];
110. $printResult = 1;
111. $found = 1;
112. break;
113. }
114. else
115. {
116. $gejala\_result = explode(',', $rule->result);
117. $gejala\_data = Gejala::where('id',$gejala\_result[0])->first();
118. $printResult = 1;
119. $command = 'ask';
120. $append = 0;
121. $found = 1;
122. break;
123. }
124. }
125. else
126. {
127. if(count($tempArrayGejala) > 1) //jika tinggal 1 yang belum di pop
128. {
129. $out = 0;
130. array\_pop($tempArrayGejala);
131. $arrayGejalaSorted = $tempArrayGejala;
132. sort($arrayGejalaSorted);
133. $stringGejalaSorted = implode(",", $arrayGejalaSorted);
134. }
135. elseif(count($tempArrayGejala) == 1)
136. {
137. $out=0;
138. }
139. }
140. }
141. if($found !=1 && count($tempArrayGejala) == 1)
142. $out = 1;
143. } /\*END OF IF RULES[0]\*/
144. else
145. {
146. if(count($tempArrayGejala) > 1) //jika tinggal 1 yang belum di pop
147. {
148. $out = 0;
149. array\_pop($tempArrayGejala);
150. $arrayGejalaSorted = $tempArrayGejala;
151. sort($arrayGejalaSorted);
152. $stringGejalaSorted = implode(",", $arrayGejalaSorted);
153. }
154. elseif(count($tempArrayGejala) == 1)
155. {
156. $out=0;
157. }
158. }
159. if($out)
160. {
161. if(count($tempArrayGejala) > 1) //jika tinggal 1 yang belum di pop
162. {
163. $out = 0;
164. array\_pop($tempArrayGejala);
165. $arrayGejalaSorted = $tempArrayGejala;
166. sort($arrayGejalaSorted);
167. $stringGejalaSorted = implode(",", $arrayGejalaSorted);
168. }
169. elseif(count($tempArrayGejala) == 1)
170. {
171. $out=0;
172. }
173. }
174. $while\_iterate\_count++;
175. }
176. return json\_encode([
177. 'command' => $command,
178. 'gejala' => $gejala\_data,
179. 'append' => $append,
180. 'append\_value' => $append\_value
181. ]);
182. }

Kode Sumber . Tanya Jawab Pada Server

#### Prediksi

1. **def** predict**():**
2. total\_attributes**=**376
3. label **=** pandas**.**read\_csv**(**"../Users/Kevin/PycharmProjects/TugasAkhir/dataset/label.csv"**)**
4. label **=** label**.**values
5. attributes **=** pandas**.**read\_csv**(**"../Users/Kevin/PycharmProjects/TugasAkhir/dataset/attributes-t5.csv"**)**
6. attributes **=** attributes**.**values
7. data **=** request**.**json
8. data\_gejala **=** data**.**get**(**"gejala"**)**
9. number **=** str**(**5**)**
10. gejala\_array **=** **[]**
11. **for** i **in** attributes**:**
12. number **=** str**(**i**[**0**])**
13. **if(**data\_gejala**.**get**(**str**(**number**))):**
14. value\_gejala **=** data\_gejala**.**get**(**number**)**
15. **else:**
16. value\_gejala **=** 0
17. gejala\_array**.**append**(**value\_gejala**)**
18. gejala\_array\_np\_dummy **=** **[]**
19. gejala\_array\_np\_dummy**.**append**(**gejala\_array**)**
20. gejala\_array\_np **=** numpy**.**array**(**gejala\_array\_np\_dummy**)**
21. seed **=** 7
22. numpy**.**random**.**seed**(**seed**)**
23. dataframe **=** pandas**.**read\_csv**(**"../Users/Kevin/PycharmProjects/TugasAkhir/dataset/classifier-training-t5.csv"**,** skipinitialspace**=True)**
24. dataset **=** dataframe**.**values
25. jumlah\_gejala **=** len**(**dataset**[**0**])** **-** 1
26. X\_train **=** dataset**[:,**0**:**jumlah\_gejala**]**
27. Y\_train **=** dataset**[:,**jumlah\_gejala**]**
28. #assigning testing
29. dataframe **=** pandas**.**read\_csv**(**"../Users/Kevin/PycharmProjects/TugasAkhir/dataset/classifier-testing-t5.csv"**,** skipinitialspace**=True)**
30. dataset **=** dataframe**.**values
31. X\_test **=** dataset**[:,**0**:**jumlah\_gejala**]**
32. Y\_test **=** dataset**[:,**jumlah\_gejala**]**
33. #label
34. label **=** pandas**.**read\_csv**(**"../Users/Kevin/PycharmProjects/TugasAkhir/dataset/label.csv"**)**
35. label **=** label**.**values
36. #Create a Gaussian Classifier
37. model **=** BernoulliNB**()**
38. # Train the model using the training sets
39. y\_pred **=** model**.**fit**(**X\_train**,** Y\_train**).**predict**(**X\_test**)**
40. class\_map **=** model**.**classes\_
41. score **=** model**.**score**(**X\_test**,** Y\_test**)**
42. predictions **=** model**.**predict\_proba**(**gejala\_array\_np**)**
43. result **=** **[]**
44. jsondata **=** '{ "result":['
45. index **=** 0
46. **for** pred **in** predictions**:**
47. top5 **=** pred**.**argsort**()[-**5**:][::-**1**]**
48. **for** item **in** top5**:**
49. labels **=** label**[**item**][**0**]**
50. result**.**append**(**HasilDiagnosis**(**labels**,**pred**[**item**]))**
51. json\_item\_string **=** '{"diagnosis":"' **+** str**(**class\_map**[**item**])** **+**'","probability":"'**+** str**(**pred**[**item**])** **+**'"}'
52. **if** index **<** 4**:**
53. json\_item\_string **=** json\_item\_string **+** ','
54. index **=** index**+**1
55. jsondata **=** jsondata **+** json\_item\_string
56. jsondata **=** jsondata **+** "]}"
57. format\_jsondata **=** json**.**dumps**(**jsondata**)**
58. **return** jsondata

Kode Sumber . Prediksi Pada Server

#### *Interface*

Pada bagian interface akan dibagi menjadi 3 potongan kode. Bagian pertama adalah *HTML*, bagian kedua adalah *Javascript* untuk tanya jawab yang terakhir adalah *javascript* untuk diagnosis penyakit.

1. @section('content')
2. <div class="container">
3. <div class="row">
4. <div class="col-xs-12 col-md-8 col-md-offset-2">
5. <div class="card">
6. <div class="card-content" style="padding-top: 50px; padding-bottom: 50px">
7. <div id="welcome" class="text-center">
8. <h4>Selamat datang di Aqeela dari Medify</h4>
9. <p>Aqeela akan membantu anda untuk mendiagnosis penyakit anda berdasarkan gejala yang anda alami. Perlu anda tahu bahwa diagnosis yang kami berikan tidak bersifat final, kami rekomendasikan anda menemui dokter untuk mendapatkan kepastian mengenai penyakit yang anda derita.</p>
10. <button class="btn btn-primary">Selanjutnya<div class="ripple-container"></div></button>
11. </div>
12. <div id="pasien" class="text-center" style="display: none">
13. <h4>Isi data berikut</h4>
14. <div class="row">
15. <div class="col-md-8 col-md-offset-2 col-xs-12">
16. <input class="form-control" type="text" id="pasien-nama" placeholder="Nama">
17. <input class="form-control" type="text" id="pasien-usia" placeholder="Usia">
18. <select class="selectpicker" data-style="btn btn-primary btn-round" title="Single Select" data-width="100%" id="pasien-gender">
19. <option value="1" selected>Laki laki</option>
20. <option value="2">Perempuan</option>
21. </select>
22. </div>
23. </div>
24. <button class="btn btn-primary">Selanjutnya<div class="ripple-container"></div></button>
25. </div>
26. <div id="gejala-first" class="text-center" style="display: none">
27. <h4>Apa gejala yang paling anda rasakan?</h4>
28. <div class="row">
29. <div class="col-md-8 col-md-offset-2 col-xs-12">
30. <select class="selectpicker" data-style="btn btn-primary btn-round" title="Single Select" data-live-search="true" data-size="7">
31. @foreach($gejalas as $gejala)
32. <option value="{{$gejala->id}}">{{$gejala->name}}</option>
33. @endforeach
34. </select>
35. </div>
36. </div>
37. <button class="btn btn-primary">Selanjutnya<div class="ripple-container"></div></button>
38. </div>
39. <div id="gejala-question" class="text-center" style="display: none;min-height: 300px">
40. <h4>Apakah anda merasakan <span id="fill-gejala"></span></h4>
41. <button class="btn btn-primary yes">Ya<div class="ripple-container"></div></button>
42. <button class="btn btn-danger no">Tidak<div class="ripple-container"></div></button>
43. </div>
44. <div id="diagnosis-warning" class="text-center" style="display: none;">
45. <h4>Setelah ini, saya akan memberikan kamu 5 diagnosis penyakit yang mungkin kamu alami</h4>
46. <p>Ingat diagnosis ini tidak bersifat final, kami rekomendasikan anda menemui dokter untuk mendapatkan kepastian mengenai penyakit yang anda derita.</p>
47. <button class="btn btn-primary">Selanjutnya<div class="ripple-container"></div></button>
48. </div>
49. <div id="diagnosis-result" class="row" style="display: none">
50. <h4 class="text-center ">Penyakit yang mungkin kamu derita</h4>
51. <div id="result" class="col-md-4 col-md-offset-4">
52. </div>
53. </div>
54. </div>
55. </div>
56. </div>
57. </div>
58. </div>
59. @endsection

Kode Sumber . Interface HTML Pada User

1 **function** setGejala**(**index**,**value**){**

2 **if(**value **==** 1**)**

3 **{**

4 count\_true**++**

5 **}**

6 count**++**

7 gejalaPasien**.**set**(**index**,**value**)**

8 **var** pasienNama **=** $**(**'#pasien-nama'**).**val**()**

9 **var** pasienUsia **=** $**(**'#pasien-usia'**).**val**()**

10 **var** pasienGender **=** $**(**'#pasien-gender'**).**val**()**

11

12 **var** gejalaPasienJSON **=** **new** Backbone**.**Model**({**

13 "gejala" **:** gejalaPasien**,**

14 "jenis\_kelamin" **:** pasienGender**,**

15 "usia" **:** pasienUsia

16 **});**

17

18 **var** gejalaPasienJSON **=**   
 JSON**.**stringify**(**gejalaPasienJSON**);**

19

20 $**.**ajax**({**

21 type**:** "POST"**,**

22 url**:** "{{url('api/app/question/next')}}"**,**

23 contentType**:** "application/json"**,**

24 headers**:** **{**

25 'Access-Control-Allow-Credentials' **:**   
 'true'

26 **},**

27 data **:** gejalaPasienJSON**,**

28 processData**:** **false,**

29 success**:** **function** **(**data**)** **{**

30 **var** obj **=** JSON**.**parse**(**data**)**

31 **var** command **=** obj**.**command

32 **var** gejala **=** obj**.**gejala

33 **var** append **=** obj**.**append

34 **var** append\_value **=** obj**.**append\_value

35 currentGejala **=** gejala**.**id

36 console**.**log**(**count**)**

37 **if(**count\_true **>** 10**)**

38 **{**

39 submitFinal**(**gejalaPasienJSON**);**

40 **}**

41 **else** **if(**count **>** 20**)**

42 **{**

43 submitFinal**(**gejalaPasienJSON**);**

44 **}**

45 **else**

46 **{**

47 **if(**command **===** 'ask'**)**

48 **{**

49 console**.**log**(**'ask'**)**

50 $**(**"#gejala-  
 question"**).**fadeOut**(**500**,** **function(){**

51 $**(**"#gejala-  
 question"**).**fadeIn**(**500**);**

52 $**(**'#gejala-question   
 #fill-gejala'**).**html**(**gejala**.**name' - '**+**   
 gejala**.**id**);**

53 **});**

54 **}**

55 **else** **if(**command **===** 'append'**)**

56 **{**

57 console**.**log**(**'append'**)**

58   
 setGejala**(**append**,**append\_value**);**

59 **}**

60 **else**

61 **{**

62 console**.**log**(**command**)**

63   
 submitFinal**(**gejalaPasienJSON**);**

64 **}**

65 **}**

66 **},**

67 error**:** **function** **(**e**)** **{**

68 **alert(**e**)**

69 **}**

70 **});**

71 **}**

Kode Sumber . *Javascript* Tanya Jawab Pada *Interface User*

1 **function** submitFinal**(**gejalaPasienJSON**)**

2 **{**

3 $**.**ajax**({**

4 type**:** "POST"**,**

5 url**:** "http://127.0.0.1:5000/predict"**,**

6 contentType**:** "application/json"**,**

7 headers**:** **{**

8 'Access-Control-Allow-Credentials' **:**   
 'true'

9 **},**

10 data **:** gejalaPasienJSON**,**

11 dataType **:** 'json'**,**

12 processData**:** **false,**

13 success**:** **function** **(**response**)** **{**

14 diagnosis **=** response**.**result**;**

15 $**(** "#diagnosis-result #result"   
 **).**html**(**''**)**

16

17 $**.**each**(**diagnosis**,** **function(**item**)** **{**

18 probability **=**   
 Number**(**diagnosis**[**item**].**probability **\***   
 100**).**toFixed**(**2**);**

19 result **=** '<h4   
 class="title">'**+**diagnosis**[**item**].**diagnosis**+** '<small>· '**+**probability**+**'%</small></h4>'

20

21 $**(** "#diagnosis-result #result" **).**append**(** result **);**

22 **})**

23

24 $**(**"#gejala-question"**).**fadeOut**(**500**,**   
 **function(){**

25 $**(**"#diagnosis-  
 warning"**).**fadeIn**(**500**);**

26 **});**

27 **},**

28 error**:** **function** **(**e**)** **{**

29 **alert(**e**);**

30 **}**

31 **});**

32

Kode Sumber . *Javascript* Diagnosis Pada User

***[Halaman ini sengaja dikosongkan]***

# BAB V PENGUJIAN DAN EVALUASI

Pada bab ini akan dijelaskan skenario dan hasil dari uji coba aplikasi yang telah diimplementasikan.

## Lingkungan Pengujian

Lingkungan pengujian yang digunakan untuk *server* dan *client* (pengguna) adalah :

Tabel . Lingkungan Pengujian *Server*

|  |  |
| --- | --- |
| **Komponen** | **Spesifikasi** |
| CPU | Intel® Core i5™ 7200U CPU @ 2.50 GHz (4 CPUs), ~ 2.7GHz |
| GPU | NVidia GeForce 720 930MX 2GB |
| Sistem Operasi | Windows 10 Pro |
| Memori | 8GB DDR4 |
| Penyimpanan | 1 TB |
| Perangkat Lunak Pendukung | * Python 3.6 * Python 2.7 * PHP 7.1 * MySQL 5.5 * Google Chrome |

Tabel . Lingkungan Pengujian *Client* (Pengguna)

|  |  |
| --- | --- |
| **Komponen** | **Spesifikasi** |
| CPU | Intel® Core i5™ 7200U CPU @ 2.50 GHz (4 CPUs), ~ 2.7GHz |
| GPU | NVidia GeForce 720 930MX 2GB |
| Sistem Operasi | Windows 10 Pro |
| Memori | 8GB DDR4 |
| Penyimpanan | 1 TB |
| Perangkat Lunak Pendukung | * Python 3.6 * Python 2.7 * PHP 7.1 * MySQL 5.5 * Google Chrome |

## Data Uji Coba

Data yang digunakan untuk uji coba aplikasi ini adalah kumpulan data kasus yang berisikan gejala gejala dan 1 diagnosis berjumlah 199 data. Dimana masing masing diagnosis (66 diagnosis) mendapatkan 3 kasus.

Adapun data rules yang digunakan pada skenario data uji coba ini sebagai berikut.

Tabel . Jumlah Data Gejala dan Rules

|  |  |  |
| --- | --- | --- |
| **Threshold** | **Jumlah Gejala** | **Rules** |
|
| **3** | 250 | 289,940 |
| **5** | 204 | 30,940 |
| **8** | 159 | 6,006 |
| **10** | 133 | 3,290 |
| **12** | 110 | 1,932 |
| **15** | 88 | 1,070 |
| **18** | 74 | 706 |
| **20** | 64 | 528 |
| **23** | 57 | 346 |

## Skenario Uji Coba

Skenario pertama yang dijalankan pada evaluasi ini berupa perhitungan akurasi dan *retrieval*. Pengujian akan dimulai dengan filter *support* untuk gejala gejala. Ada 9 *minimum support* yang akan di filter (3, 5, 8, 10, 12, 15, 18, 20, 23). Setelah filter, data testing gejala ini akan di ujikan kepada model *classifer*. Dan hasilnya akan dicatat pada file teks.

Skenario kedua adalah menguji coba aplikasi kepada pengguna. Dengan batasan pertanyaan hingga 30, pengguna akan menggunakan aplikasi hingga pengguna merasa jenuh. Jika pengguna telah merasa jenuh, pengguna dapat memberitahu dan diizinkan untuk berhenti melakukan pengujian aplikasi. Tujuan dari skenario ini adalah mencari jumlah pertanyaan yang tidak menyebabkan pengguna bosan hingga berakibat keluar dari aplikasi sebelum muncul diagnosis dari aplikasi.

Pada skenario kedua ini uji pengguna dilakukan oleh masyarakat awan yang berusia 20 – 25 tahun. Masing masing mencoba aplikasi 1 kali, setiap pengguna diberikan 1 diagnosis penyakit yang populer pada skenario ini yang digunakan adalah demam berdarah, dan ankilostomiasis (cacing tambang). Jika pengguna tidak mengetahui banyak mengenai diagnosis tersebut maka akan diberikan sedikit gambaran dan daftar gejala yang mungkin. Dan setelah diberikan penjelasan dan daftar gejala pengguna bebas untuk mengisi berdasarkan apa yang dibaca atau pengalaman yang dia miliki.

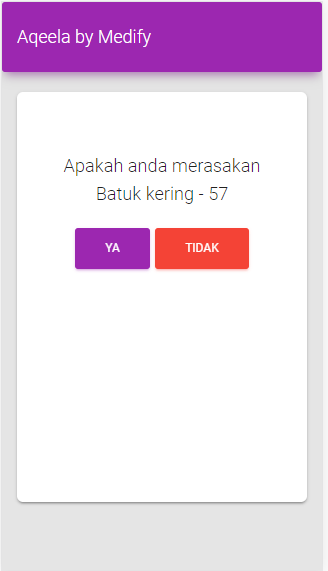
Selain itu aplikasi juga akan diujikan kepada dokter untuk menilai bagaimana pendapat dokter mengenai akurasi dari pemetaan gejala dan juga hasil diagnosis dokter. Pada uji coba ini dokter akan memasukkan gejala awal dan menjawab pertanyaan, lalu dokter akan menyimpulakan apakah diagnosis yang diberikan masuk akal dengan hasil yang diberikan oleh aplikasi. Pada pengujian ini dokter akan melakukan pengujian aplikasi sebanyak 10 kali.

## Hasil Uji Coba

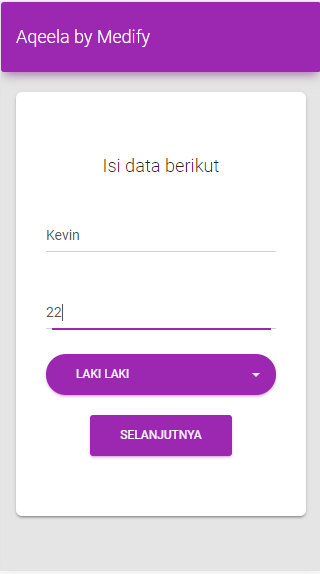
Pada sub bab ini akan dijelaskan bagaimana hasil dari hasil uji coba dari evaluasi secara sistem dan juga evaluasi oleh dokter. Adapun berikut adalah tampilan dari aplikasi yang akan digunakan untuk pengujian.



Gambar .3 Tampilan Tanya Jawab



Gambar .2 Tampilan Hasil Diagnosis



Gambar .1 Pengisian Data Pasien

### Hasil Uji Sistem

Hasil uji skenario sistem akan dibagi menjadi 2 jenis evaluasi. Evaluasi secara akurasi dan evaluasi secara *information retrieval.*

Pada uji coba berdasarkan akurasi ini dibagi menjadi 2 *classifer* yaitu *Naïve Bayes* dan SVM. Hasilnya dapat dilihat melalui tabel dan grafik dibawah ini.

Tabel . Hasil Uji Akurasi

|  |  |  |
| --- | --- | --- |
| **Min Support** | **Naïve Bayes** | **SVM** |
| **3** | 75.897 | 75.897 |
| **5** | 76.381 | 75.376 |
| **8** | 74.371 | 73.366 |
| **10** | 74.874 | 71.859 |
| **12** | 71.859 | 71.356 |
| **15** | 71.859 | 69.849 |
| **18** | 63.316 | 60.301 |
| **20** | 60.301 | 57.788 |
| **23** | 56.281 | 51.256 |

Gambar . Grafik Hasil Uji Akurasi

Pada hasil akurasi dapat dilihat bahwa grafik cenderung menurun walaupun dengan selisih yang sangat kecil. Pada saat *minimum support* bernilai 5, *Naïve Bayes* memiliki akurasi tertinggi sedangkan SVM memiliki akurasi tertinggi pada saat *minimum support* bernilai 3.

Selisih pada setiap hasil akurasi juga berbanding tipis antar *minimum support*. Rata rata selisih hasil akurasi classifier pada setiap *minimum support* adalah 2,69 untuk *Naïve Bayes* dan 3,08 untuk SVM. Selisih terbesar antar support terjadi saat *minimum support* 15 dibandingkan dengan *minimum support* 18, selisih antar kedua hasil ini adalah 8,5% untuk Naïve Bayes dan 9,5% untuk SVM. Adapun rata rata selisih hasil akurasi antar *minimum support* pada saat *minimum support* diantara 3 – 15 adalah 1.2% untuk *Naïve Bayes* dan SVM. Selisih dari hasil tertinggi dengan terendah adalah 20,1% untuk *Naïve bayes* dan 24,6% untuk SVM.

Tabel . Hasil Uji Retrieval

|  |  |  |
| --- | --- | --- |
| **Min Support** | **Naïve Bayes** | **SVM** |
| **3** | 188 | 187 |
| **5** | 189 | 184 |
| **8** | 190 | 186 |
| **10** | 190 | 187 |
| **12** | 190 | 190 |
| **15** | 186 | 185 |
| **18** | 184 | 179 |
| **20** | 180 | 177 |
| **23** | 176 | 175 |

Gambar . Grafik Hasil Uji Retrieval

Pada uji *retrieval* skor tertinggi adalah 190 yang dimiliki *Naïve Bayes* saat *minimum support* bernilai 8, 10, dan 12. Skor 190 juga dicatatkan oleh SVM pada saat *minimum support* bernilai 12. Selisih rata rata *retrieval* antar *minimum support* pada Naïve Bayes bernilai 2 sedangkan pada SVM bernilai 3. Sedangkan saat *minimum support* bernilai 3 hingga 15 Naïve Bayes memiliki skor rata rata selisih 1.2 sedangkan SVM 2.8. Pada Naïve Bayes penurunan skor tertinggi terjadi ketika saat *minimum support* ke 12 menuju 15, 18 menuju 20 dan 20 menuju 23. Sedangkan pada SVM penurunan skor tertinggi terjadi saat *minimum* *support* ke 15 menuju 18.

Tabel . Hasil Uji Peringkat

|  |  |  |
| --- | --- | --- |
| **Min Support** | **Naïve Bayes** | **SVM** |
| **3** | 87.135 | 85.829 |
| **5** | 87.738 | 85.829 |
| **8** | 87.738 | 86.030 |
| **10** | 88.040 | 85.326 |
| **12** | 87.738 | 85.628 |
| **15** | 86.633 | 82.512 |
| **18** | 82.512 | 78.894 |
| **20** | 80.201 | 76.884 |
| **23** | 76.683 | 73.969 |

Gambar . Grafik Hasil Uji Peringkat

Pada uji peringkat nilai tertinggi adalah 88,04 yang dimiliki Naïve Bayes saat *minimum support* bernilai 10. SVM memiliki nilai tertinggi sebesar 85.829. Selisih rata rata nilai peringkat antar *minimum support* pada Naïve Bayes bernilai 1.53 sedangkan pada SVM bernilai 2.6. Sedangkan saat *minimum support* bernilai 3 hingga 15 *Naïve Bayes* memiliki skor rata rata selisih 0.46 sedangkan SVM 1.95. Pada Naïve Bayes penurunan skor tertinggi terjadi ketika saat *minimum support* ke 15 menuju 18. SVM mengalami penurunan skor tertinggi terjadi juga saat minimum support ke 15 menuju 18.

### Hasil Uji Pengguna

Uji pengguna dilakukan oleh 10 orang dengan hasil sebagai berikut.

Tabel . Hasil Uji Pengguna Dengan Pertanyaan Maksimal 30

|  |  |
| --- | --- |
| **Pengguna** | **Pertanyaan ke** |
| 1 | 20 |
| 2 | 22 |
| 3 | 18 |
| 4 | 15 |
| 5 | 18 |
| 6 | 23 |
| 7 | 22 |
| 8 | 23 |
| 9 | 20 |
| 10 | 17 |

Berdasarkan hasil tersebut, karena tidak ada pengguna yang menggunakan aplikasi hingga selesai maka uji coba dilaksanakan ulang terhadap pengguna yang sama dengan jumlah pertanyaan sebanyak 20. Dengan hasil sebagai berikut

Tabel . Hasil Uji Pengguna Dengan Pertanyaan Maksimal 20

|  |  |
| --- | --- |
| **Pengguna** | **Pertanyaan ke** |
| 1 | 20 |
| 2 | 20 |
| 3 | 20 |
| 4 | 20 |
| 5 | 20 |
| 6 | 20 |
| 7 | 20 |
| 8 | 20 |
| 9 | 18 |
| 10 | 18 |

Dapat dilihat pada hasil kedua 80% pengguna menyelesaikan aplikasi hingga memunculkan diagnosis. Kebanyakan pengguna berkomentar jika ada gejala yang tidak berhubungan, dan merasa bosan jika gejala tidak berhubungan. Tetapi jika gejala saling terkait dan mereka menjawab Ya maka keinginan mereka untuk menjawab pertanyaan berikutnya menjadi lebih tinggi.

### Hasil Uji Dokter

Uji pengguna dilakukan oleh dokter yang langsung mencoba aplikasi. Dari 10 percobaan yang dilakukan oleh dokter. Dokter menilai bahwa aplikasi cukup baik walau diagnosis yang diberikan terkadang sedikit meleset. Dari segi pertanyaan dokter menilai sudah relevan dan saling terkait. Berikut merupakan salah satu hasil uji coba yang dilakukan dokter:

Data pasien:

Kevin, Laki Laki, 22 Tahun

|  |  |
| --- | --- |
| **Gejala** | **Jawaban** |
| Sakit perut | Ya (jawaban awal) |
| Muntah | Ya |
| Mual | Ya |
| Demam | Tidak |
| Diare | Tidak |
| Nafsu makan berkurang | Tidak |
| Malaise | Tidak |
| Kepala sakit | Ya |
| Demam tinggi | Tidak |
| Otot nyeri | Tidak |
| Buang air besar berdarah | Tidak |
| Perut kram | Ya |
| Kulit bintik bintik merah | Tidak |
| Perdarah gusi | Tidak |
| Kulit ruam | Tidak |
| Batuk | Ya |
| Batuk kering | Tidak |
| Batuk berdahak | Tidak |
| Tenggorokan Sakit | Ya |
| Hidung mengeluarkan ingus | Ya |
| Bersin | Ya |

Hasil diagnosis :

faringitis· 92.24%

influenza· 3.00%

rhinitis· 2.14%

leptospirosis· 0.83%

ankilostomiasis· 0.41%

## Analisis Hasil Uji Coba

Berdasarkan hasil uji coba pada sistem, *Naïve Bayes* memberikan hasil yang lebih baik daripada SVM. Peningkatan *minimum support* secara umum menyebabkan akurasi *classifier* turun. Hal ini dikarenakan gejala yang semakin sedikit menyebabkan kurangnya variasi fitur dari data.

Selain itu data dengan akurasi yang tertinggi tidak menyebabkan skor *retrieval* maupun peringkat juga menjadi tertinggi. Hal ini mungkin disebabkan karena sistem peringkat, karena tidak munculnya diagnosis pada 5 diagnosis tertinggi menyebabkan skor peringkat bernilai 0. Walaupun selisih nya sedikit, data dengan *classifier* Naïve Bayes dan *minimum support* sebesar 3 memberikan performa yang terbaik.

Jika diperhatikan hasil retrieval pada uji coba ini, artinya ada diagnosis yang tidak muncul pada uji coba data uji. Dari analisis dataset dapat diperkirakan penyebabnya adalah banyaknya diagnosis yang memiliki gejala sama dengan diagnosis lainnya, terutama pada diagnosis mata, kardiovaskular, dan pernapasan.

Pada hasil uji coba pengguna dapat dilihat bahwa terlalu banyak pertanyaan menyebabkan pengguna jenuh. Walaupun secara teori akan memberikan hasil retrieval yang lebih baik karena fitur semakin banyak dan spesifik, tetapi jika pengguna enggan menyelesaikan maka tujuan dari aplikasi ini juga tidak dipenuhi.

Pengguna mengeluhkan adanya ketidaksesuaian pertanyaan gejala dengan pertanyaan sebelumnya dapat disebabkan oleh beberapa hal berikut:

* Aplikasi memberikan gejala yang meluas untuk sebelum memberika gejala yang spesifik. Sedangkan kebanyakan pengguna langsung menginginkan pertanyaan yang spesifik berdasarkan pengetahuan dia mengenai penyakit tersebut.
* Banyak data pada sumber data yang memberikan daftar gejala berbeda dengan yang pernah dialami pengguna. Sehingga pengguna merasa gejala tersebut tidak relevan.
* Pengguna merasa pertanyaan diulang, karena ada banyak gejala mirip tapi tak sama. Seperti perut nyeri dengan mual, sakit kepala dan pusing.

Pada uji dokter, dari penilaian dokter sudah memberikan respon positif baik dari segi hasil diagnosis maupun relevansi antar pertanyaan. Walau ada beberapa diagnosis yang salah, umumnya diagnosis yang salah tersebut menempati peringkat 4 dan 5. Sehingga tidak terlalu mengganggu hasil diagnosis.

# BAB VI KESIMPULAN DAN SARAN

Pada bab ini akan diberikan hasil kesimpulan yang diperoleh dari tugas akhir yang telah dikerjakan dan saran tentang pengembangan dari tugas akhir ini yang dapat dilakukan di masa yang akan datang.

## Kesimpulan

Kesimpulan yang diperoleh berdasarkan hasil uji coba dan evaluasi tugas akhir ini adalah :

* + - 1. FP Tree memiliki hasil yang cukup baik dalam memetakan atribut atribut yang berkaitan.
      2. Filter *support* pada FP Tree memberikan hasil yang cukup signifikan pada pengalaman pengguna dan juga hasil diagnosis yang diberikan.
      3. Berdasarkan hasil pengujian *Naïve Bayes* dan SVM memberikan hasil yang hampir sama. Tetapi *Naïve Bayes* memberikan hasil yang lebih baik daripada SVM pada saat *minimum support* bernilai 5 dengan akurasi 76,381%, skor IR 87,738% dan IR *Show* 189. Hal ini membuktikan *minimum support* berpengaruh pada performa *classifier*.

## Saran

Saran yang diberikan untuk pengembangan aplikasi ini kedepannya adalah :

Memperbanyak variasi data kasus dan diagnosis, agar gejala yang ditanyakan lebih bervariatif dan hasil diagnosis yang lebih tepat. Data kasus juga didasarkan pada data lapangan yang mungkin di masa depan bisa didapatkan melalui lembaga pemerintah atau rumah sakit. Perkiraan agar data menjadi lebih baik adalah sekitar 100 kasus untuk setiap diagnosis.

Mengubah *classifier* menjadi *Deep Learning* agar memberikan hasil dan performa yang lebih baik.

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | A. Kirk, "The Telegraph," Telegraph Media Group, 24 July 2015. [Online]. Available: https://www.telegraph.co.uk/news/health/news/11760658/One-in-four-self-diagnose-on-the-internet-instead-of-visiting-the-doctor.html. [Accessed 2018 May 27]. |
| [2] | ecisapare, "Quizlet," 2014. [Online]. Available: https://quizlet.com/34170376/medical-terminology-medical-terms-for-disease-diagnosis-treatment-flash-cards/. [Accessed 1 January 2018]. |
| [3] | Unknown, "Medical Bits," 5 January 2014. [Online]. Available: https://medicalbits.wordpress.com/2014/01/05/the-anamnesis-taking-a-medical-history/. [Accessed 8 January 2018]. |
| [4] | A. Maae, "Quora," Quora, 25 April 2017. [Online]. Available: https://www.quora.com/What-is-support-and-confidence-in-data-mining/answer/Azim-Maae?srid=3QjiQ. [Accessed 1 January 2018]. |
| [5] | Unknown, "WikiBooks," 6 August 2017. [Online]. Available: https://en.wikibooks.org/wiki/Data\_Mining\_Algorithms\_In\_R/Frequent\_Pattern\_Mining/The\_FP-Growth\_Algorithm. [Accessed 8 January 2018]. |
| [6] | M. S. S. S. Tina R. Patil, "Performance Analysis of Naive Bayes and J48," *International Journal Of Computer Science And Applications,* vol. 6, p. 2, 2013. |
| [7] | B. Stecanella, "MonkeyLearn," MonkeyLearn Inc., 25 May 2017. [Online]. Available: https://monkeylearn.com/blog/practical-explanation-naive-bayes-classifier/. [Accessed 30 May 2018]. |
| [8] | S. Patel, "Medium," Medium Inc, 3 May 2017. [Online]. Available: https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72. [Accessed 30 May 2018]. |
| [9] | B. Stecanella, "MonkeyLearn," MonkeyLearn Inc, 22 June 2017. [Online]. Available: https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/. [Accessed 30 May 2018]. |
| [10] | Unknown, "ScikitLearn - Wikipedia," Wikipedia, 16 April 2018. [Online]. Available: https://en.wikipedia.org/wiki/Scikit-learn. [Accessed 30 May 2018]. |
| [11] | A. Ronacher, "Flask," - - 2010-2018. [Online]. Available: http://flask.pocoo.org/. [Accessed 30 May 2018]. |
| [12] | A. Ronacher, "Flask," Flask, 2013. [Online]. Available: https://web.archive.org/web/20171117015927/http://flask.pocoo.org:80/docs/0.10/foreword. [Accessed 30 May 2018]. |
| [13] | Laravel, "Github," Github Inc, 19 May 2018. [Online]. Available: https://github.com/laravel/framework. [Accessed 30 May 2018]. |
| [14] | J. P. a. Y. Y. Jiawei Han, "Mining Frequent Patterns without Candidate Generation". |

# LAMPIRAN

## Hasil Lengkap Uji Sistem

Tabel . Hasil Pengujian *Naive Bayes*, *Min Support* = 5

|  |  |  |
| --- | --- | --- |
| TRUE | Prediksi |  |
| rhinitis | rhinitis | ✓ |
| rhinitis | rhinitis | ✓ |
| rhinitis | rhinitis | ✓ |
| influenza | influenza | ✓ |
| influenza | influenza | ✓ |
| influenza | influenza | ✓ |
| faringitis | influenza |  |
| faringitis | influenza |  |
| faringitis | leptospirosis |  |
| tonsilitis | tonsilitis | ✓ |
| tonsilitis | tonsilitis | ✓ |
| tonsilitis | tonsilitis | ✓ |
| laringitis | laringitis | ✓ |
| laringitis | laringitis | ✓ |
| laringitis | laringitis | ✓ |
| pneumonia | tuberkulosis |  |
| pneumonia | pneumonia | ✓ |
| pneumonia | pneumonia | ✓ |
| bronkopneumonia | bronkopneumonia | ✓ |
| bronkopneumonia | pneumotoraks |  |
| bronkopneumonia | pneumonia |  |
| pneumotoraks | pneumotoraks | ✓ |
| pneumotoraks | pneumotoraks | ✓ |
| pneumotoraks | pneumotoraks | ✓ |
| ppok-penyakit-paru-obstruktif-kronis | ppok-penyakit-paru-obstruktif-kronis | ✓ |
| ppok-penyakit-paru-obstruktif-kronis | ppok-penyakit-paru-obstruktif-kronis | ✓ |
| ppok-penyakit-paru-obstruktif-kronis | pneumotoraks |  |
| epistaksis | epistaksis | ✓ |
| epistaksis | epistaksis | ✓ |
| epistaksis | epistaksis | ✓ |
| sinusitis | sinusitis | ✓ |
| sinusitis | sinusitis | ✓ |
| sinusitis | sinusitis | ✓ |
| tuberkulosis | tuberkulosis | ✓ |
| tuberkulosis | tuberkulosis | ✓ |
| tuberkulosis | tuberkulosis | ✓ |
| morbili | morbili | ✓ |
| morbili | morbili | ✓ |
| morbili | morbili | ✓ |
| varisela | varisela | ✓ |
| varisela | varisela | ✓ |
| varisela | varisela | ✓ |
| malaria | malaria | ✓ |
| malaria | malaria | ✓ |
| malaria | malaria | ✓ |
| leptospirosis | leptospirosis | ✓ |
| leptospirosis | leptospirosis | ✓ |
| leptospirosis | leptospirosis | ✓ |
| filariasis | filariasis | ✓ |
| filariasis | filariasis | ✓ |
| filariasis | filariasis | ✓ |
| lepra | perdarahan-subkonjungtiva |  |
| lepra | lepra | ✓ |
| lepra | lepra | ✓ |
| lepra | lepra | ✓ |
| keracunan-makanan | keracunan-makanan | ✓ |
| keracunan-makanan | keracunan-makanan | ✓ |
| keracunan-makanan | kolesistitis |  |
| alergi-makanan | alergi-makanan | ✓ |
| alergi-makanan | ankilostomiasis |  |
| alergi-makanan | perdarahan-subkonjungtiva |  |
| demam-berdarah-dengue | leptospirosis |  |
| demam-berdarah-dengue | demam-berdarah-dengue | ✓ |
| demam-berdarah-dengue | kolesistitis |  |
| anemia | anemia | ✓ |
| anemia | anemia | ✓ |
| anemia | anemia | ✓ |
| hiv-aids | hiv-aids | ✓ |
| hiv-aids | lupus-eritematosus-sistemik |  |
| hiv-aids | keracunan-makanan |  |
| lupus-eritematosus-sistemik | lupus-eritematosus-sistemik | ✓ |
| lupus-eritematosus-sistemik | lupus-eritematosus-sistemik | ✓ |
| lupus-eritematosus-sistemik | lupus-eritematosus-sistemik | ✓ |
| limfadenitis | limfadenitis | ✓ |
| limfadenitis | peritonitis |  |
| limfadenitis | limfadenitis | ✓ |
| asam-lambung | asam-lambung | ✓ |
| asam-lambung | asam-lambung | ✓ |
| asam-lambung | asam-lambung | ✓ |
| hipertensi | pneumotoraks |  |
| hipertensi | hipertensi | ✓ |
| hipertensi | takikardia |  |
| cardiorespiratory-arrest | cardiorespiratory-arrest | ✓ |
| cardiorespiratory-arrest | cardiorespiratory-arrest | ✓ |
| cardiorespiratory-arrest | cardiorespiratory-arrest | ✓ |
| gagal-jantung | gagal-jantung | ✓ |
| gagal-jantung | gagal-jantung | ✓ |
| gagal-jantung | ppok-penyakit-paru-obstruktif-kronis |  |
| takikardia | takikardia | ✓ |
| gastritis | kolesistitis |  |
| gastritis | gastritis | ✓ |
| takikardia | infark-miokard |  |
| gastritis | gastritis | ✓ |
| takikardia | takikardia | ✓ |
| infark-miokard | infark-miokard | ✓ |
| gastroenteritis | malaria |  |
| gastroenteritis | keracunan-makanan |  |
| gastroenteritis | keracunan-makanan |  |
| disentri | disentri | ✓ |
| disentri | disentri | ✓ |
| disentri | disentri | ✓ |
| hemoroid | hemoroid | ✓ |
| infark-miokard | infark-miokard | ✓ |
| hemoroid | hemoroid | ✓ |
| hemoroid | hemoroid | ✓ |
| infark-miokard | angina-pektoris-stabil |  |
| angina-pektoris-stabil | infark-miokard |  |
| hepatitis-a | hepatitis-b |  |
| hepatitis-a | hepatitis-a | ✓ |
| hepatitis-a | hepatitis-b |  |
| angina-pektoris-stabil | angina-pektoris-stabil | ✓ |
| angina-pektoris-stabil | angina-pektoris-stabil | ✓ |
| hepatitis-b | hepatitis-b | ✓ |
| hepatitis-b | hepatitis-a |  |
| hepatitis-b | hepatitis-b | ✓ |
| kolesistitis | kolesistitis | ✓ |
| kolesistitis | kolesistitis | ✓ |
| kolesistitis | kolesistitis | ✓ |
| apendisitis | peritonitis |  |
| apendisitis | peritonitis |  |
| apendisitis | peritonitis |  |
| peritonitis | peritonitis | ✓ |
| peritonitis | peritonitis | ✓ |
| peritonitis | peritonitis | ✓ |
| serumen-prop | serumen-prop | ✓ |
| serumen-prop | serumen-prop | ✓ |
| serumen-prop | serumen-prop | ✓ |
| benda-asing-di-telinga | benda-asing-di-telinga | ✓ |
| benda-asing-di-telinga | benda-asing-di-telinga | ✓ |
| benda-asing-di-telinga | benda-asing-di-telinga | ✓ |
| askariasis | askariasis | ✓ |
| askariasis | askariasis | ✓ |
| askariasis | askariasis | ✓ |
| ankilostomiasis | ankilostomiasis | ✓ |
| ankilostomiasis | ankilostomiasis | ✓ |
| ankilostomiasis | ankilostomiasis | ✓ |
| skistosomiasis | skistosomiasis | ✓ |
| skistosomiasis | skistosomiasis | ✓ |
| skistosomiasis | skistosomiasis | ✓ |
| taeniasis | taeniasis | ✓ |
| taeniasis | taeniasis | ✓ |
| taeniasis | taeniasis | ✓ |
| strongiloidiasis | strongiloidiasis | ✓ |
| strongiloidiasis | strongiloidiasis | ✓ |
| strongiloidiasis | strongiloidiasis | ✓ |
| mata-kering | trikiasis |  |
| mata-kering | perdarahan-subkonjungtiva |  |
| mata-kering | konjungtivitis |  |
| buta-senja | buta-senja | ✓ |
| buta-senja | buta-senja | ✓ |
| buta-senja | buta-senja | ✓ |
| hordeolum-bintitan | hordeolum-bintitan | ✓ |
| hordeolum-bintitan | hordeolum-bintitan | ✓ |
| hordeolum-bintitan | hordeolum-bintitan | ✓ |
| konjungtivitis | konjungtivitis | ✓ |
| konjungtivitis | hordeolum-bintitan |  |
| konjungtivitis | konjungtivitis | ✓ |
| blefaritis | konjungtivitis |  |
| blefaritis | blefaritis | ✓ |
| blefaritis | blefaritis | ✓ |
| perdarahan-subkonjungtiva | trikiasis |  |
| perdarahan-subkonjungtiva | perdarahan-subkonjungtiva | ✓ |
| perdarahan-subkonjungtiva | perdarahan-subkonjungtiva | ✓ |
| astigmatisme | astigmatisme | ✓ |
| astigmatisme | astigmatisme | ✓ |
| astigmatisme | astigmatisme | ✓ |
| hipermetropia | astigmatisme |  |
| hipermetropia | hipermetropia | ✓ |
| hipermetropia | hipermetropia | ✓ |
| miopia-ringan | miopia-ringan | ✓ |
| miopia-ringan | miopia-ringan | ✓ |
| miopia-ringan | miopia-ringan | ✓ |
| presbiopia | astigmatisme |  |
| presbiopia | astigmatisme |  |
| presbiopia | presbiopia | ✓ |
| katarak | presbiopia |  |
| katarak | retinopati-diabetik |  |
| katarak | katarak | ✓ |
| glaukoma | glaukoma | ✓ |
| glaukoma | glaukoma | ✓ |
| glaukoma | glaukoma | ✓ |
| trikiasis | trikiasis | ✓ |
| trikiasis | blefaritis |  |
| trikiasis | episkleritis |  |
| episkleritis | episkleritis | ✓ |
| episkleritis | episkleritis | ✓ |
| episkleritis | episkleritis | ✓ |
| hifema | hifema | ✓ |
| hifema | hifema | ✓ |
| hifema | hifema | ✓ |
| retinopati-diabetik | retinopati-diabetik | ✓ |
| retinopati-diabetik | retinopati-diabetik | ✓ |
| retinopati-diabetik | hifema |  |
| otitis-eksterna | otitis-eksterna | ✓ |
| otitis-eksterna | otitis-eksterna | ✓ |
| otitis-eksterna | otitis-eksterna | ✓ |
| otitis-media | otitis-media | ✓ |
| otitis-media | otitis-media | ✓ |
| otitis-media | otitis-eksterna |  |

## 2. Hasil Lengkap Uji Dokter

### Uji Coba 1

Laki Laki, 22 Tahun

|  |  |
| --- | --- |
| **Gejala** | **Jawaban** |
| Sakit perut | Ya (jawaban awal) |
| Muntah | Ya |
| Mual | Ya |
| Demam | Tidak |
| Diare | Tidak |
| Nafsu makan berkurang | Tidak |
| Malaise | Tidak |
| Kepala sakit | Ya |
| Demam tinggi | Tidak |
| Otot nyeri | Tidak |
| Buang air besar berdarah | Tidak |
| Perut kram | Ya |
| Kulit bintik bintik merah | Tidak |
| Perdarah gusi | Tidak |
| Kulit ruam | Tidak |
| Batuk | Ya |
| Batuk kering | Tidak |
| Batuk berdahak | Tidak |
| Tenggorokan Sakit | Ya |
| Hidung mengeluarkan ingus | Ya |
| Bersin | Ya |

Hasil diagnosis :

faringitis· 92.24%

influenza· 3.00%

rhinitis· 2.14%

leptospirosis· 0.83%

ankilostomiasis· 0.41%

### Uji Coba 2

Laki Laki, 62 Tahun

|  |  |
| --- | --- |
| **Gejala** | **Jawaban** |
| Penglihatan Kabur | Ya (jawaban awal) |
| Kepala Sakit/Nyeri | Tidak |
| memincingkan mata untuk melihat lebih jelas | Ya |
| Penglihatan kabur jarak jauh | Ya |
| Mata lelah | Ya |
| Pusing | Tidak |
| Mata perih mata panas | Tidak |
| penglihatan kabur jarak dekat | Ya |
| Mata kering | Tidak |
| Mata gatal | Tidak |
| Fotofobia | Tidak |
| kesulitan membaca | Ya |
| Mata merah | Tidak |

Hasil diagnosis :

miopia-ringan· 52.41%

hipermetropia· 45.48%

astigmatisme· 1.91%

presbiopia· 0.21%

retinopati-diabetik· 0.00%

### Uji Coba 3

Perempuan, 22 Tahun

|  |  |
| --- | --- |
| **Gejala** | **Jawaban** |
| Sakit Perut | Ya (jawaban awal) |
| Muntah | Tidak |
| Mual | Tidak |
| Diare | Tidak |
| Demam | Tidak |
| Nafsu Makan Berkurang | Tidak |
| Malaise | Ya |
| Penurunan berat badan | Tidak |
| Ikterus | Tidak |
| Menggigil | Tidak |
| Batuk | Tidak |
| Kepala Sakit/Nyeri | Tidak |
| Buang air besar berdarah | Ya |
| Urine berdarah | Tidak |
| kulit ruam | Tidak |
| Kulit gatal | Tidak |
| Perut kram | Ya |
| Buang air besar berulang | Ya |
| Demam Tinggi | Tidak |
| Kembung | Ya |
| Nyeri tekan perut | Ya |

Hasil diagnosis :

gastroenteritis· 71.45%

disentri· 25.36%

taeniasis· 0.67%

peritonitis· 0.53%

skistosomiasis· 0.51%

### Uji Coba 4

Laki laki, 34 Tahun

|  |  |
| --- | --- |
| **Gejala** | **Jawaban** |
| Sesak Dada | Ya |
| Dada nyeri | Ya |
| Sesak nafas | Ya |
| Nadi cepat jantung cepat | Ya |
| takipnea | Ya |
| mual | Tidak |
| Dada terasa berat | Ya |
| Batuk | Tidak |
| Malaise | Tidak |
| Demam | Tidak |
| Batuk berdahak | Tidak |
| Pusing | Tidak |
| Kepala sakit/nyeri | Tidak |
| Palpitasi | Ya |
| Kulit Ruam | Tidak |
| Tenggorokan Sakit Nyeri | Tidak |
| Merokok | Ya |
| Obesitas | Tidak |
| Mengi | Ya |
| Riwayat gangguan jantung | Ya |
| Batuk kronis | Tidak |

Hasil diagnosis :

pneumotoraks· 83.33%

infark-miokard· 12.60%

gagal-jantung· 2.80%

angina-pektoris-stabil· 0.64%

takikardia· 0.60%

### Uji Coba 5

Perempuan, 15 Tahun

|  |  |
| --- | --- |
| **Gejala** | **Jawaban** |
| Hidung tersumbat | Ya |
| Hidung mengeluarkan ingus (pilek) | Ya |
| Bersin | Ya |
| Hidung gatal | Tidak |
| Kepala sakit/nyeri | Tidak |
| Batuk | Ya |
| Batuk kering | Tidak |
| Batuk berdahak | Ya |
| Demam | Ya |
| Demam Tinggi | Tidak |
| Demam > 3 hari | Tidak |
| Tenggorokan sakit/nyeri | Ya |
| Malaise | Ya |
| Wajah nyeri | Ya |
| Kulit bintik bintik merah | Tidak |
| Ingus berwarna hijau | Tidak |
| Otot nyeri | Tidak |
| Nafas Bau | Tidak |
| Sesak nafas | Tidak |
| Nafsu makan berkurang | Ya |
| Mengigil | Tidak |

Hasil diagnosis :

influenza· 69.48%

rhinitis· 20.35%

faringitis· 4.81%

sinusitis· 3.79%

bronkopneumonia· 1.03%

### Uji Coba 6

Laki laki, 34 tahun

|  |  |
| --- | --- |
| **Gejala** | **Jawaban** |
| Suara serak | Ya |
| Tenggorokan Sakit/Nyeri | Ya |
| Batuk | Tidak |
| Demam | Tidak |
| Suara hilang | Ya |
| Batuk kering | Tidak |
| Suara parau | Ya |
| Kepala sakit/nyeri | Tidak |
| Malaise | Tidak |
| Hidung mengeluarkan ingus | Tidak |
| Tenggorokan Sakit/Nyeri saat menelan | Ya |
| Otot nyeri | Tidak |
| Nafsu makan berkurang | Tidak |
| Mual | Tidak |
| Demam tinggi | Tidak |
| Sendi nyeri | Tidak |
| Bersin | TIdak |
| Diare | Tidak |
| Sesak nafas | Tidak |
| Muntah | Tidak |
| Kulit ruam | Tidak |

Hasil diagnosis :

laringitis· 99.35%

asam-lambung· 0.41%

tonsilitis· 0.16%

perdarahan-subkonjungtiva· 0.02%

faringitis· 0.02%

### Uji Coba 7

Laki laki, 35 tahun

|  |  |
| --- | --- |
| **Gejala** | **Jawaban** |
| Demam tinggi | Ya |
| Demam | Ya |
| Demam > 3 hari | Ya |
| Kepala sakit/nyeri | Ya |
| Muntah | Tidak |
| Mual | TIdak |
| Otot Nyeri | Tidak |
| Tenggorokan Sakit/Nyeri | Tidak |
| Diare | Tidak |
| Malaise | Ya |
| Tenggorokan Sakit/Nyeri saat menelan | Tidak |
| Batuk | Ya |
| Batuk Kering | Ya |
| Hidung mengeluarkan ingus (pilek) | Tidak |
| Sesak nafas | Ya |
| Batuk berdahak | Ya |
| Nafsu makan berkurang | Ya |
| Mengigil | Tidak |
| Dada nyeri | Tidak |
| Napas Pendek dan terengah engah (napas cepat) | Tidak |
| Penurunan berat badan | Tidak |

Hasil diagnosis :

pneumonia· 59.65%

bronkopneumonia· 35.01%

influenza· 1.82%

tuberkulosis· 1.60%

ppok-penyakit-paru-obstruktif-kronis· 1.51%

### Uji Coba 8

Perempuan, 22 tahun

|  |  |
| --- | --- |
| **Gejala** | **Jawaban** |
| Pingsan mendadak | Ya |
| Tidak ada nafas | Ya |
| Pusing | Tidak |
| Henti Jantung | Ya |
| Tidak sadar | Ya |
| Dada nyeri | Ya |
| Palpitasi (Denyut Detak Jantung Tidak Beraturan) | Ya |
| Takipnea | Ya |
| Malaise | Tidak |
| Sesak nafas | Ya |
| Nadi cepat jantung cepat | Ya |
| Sesak dada | Ya |
| Mual | Tidak |
| Dada terasa berat | Tidak |
| Dada nyeri menjalar ke punggung, lengan, bahu | Ya |

Hasil diagnosis :

cardiorespiratory-arrest· 98.17%

pneumotoraks· 1.34%

infark-miokard· 0.28%

takikardia· 0.13%

angina-pektoris-stabil· 0.08%

### Uji Coba 9

Perempuan, 57 tahun

|  |  |
| --- | --- |
| **Gejala** | **Jawaban** |
| Mata merah | Ya |
| Mata berair | Ya |
| Fotofobia | Ya |
| Mata gatal | Ya |
| Mata kering | Tidak |
| Mata perih | Tidak |
| Mata bengkak | Tidak |
| Memiliki riwayat alergi | Tidak |
| Kelopak mata bengkak | Tidak |
| Kulit gatal | Tidak |
| Bersin | Tidak |
| Mata nyeri | Ya |
| Bulu mata lengket | Tidak |
| Mata sensasi berpasir | Tidak |
| Mata lelah | Tidak |
| Hidung mengeluarkan ingus (pilek) | Tidak |
| Kulit ruam | Tidak |
| Sesak nafas | Tidak |
| Mata muncul kotoran di sekeliling kelopak mata (belek) | Ya |
| Takipnea | Tidak |
| Hidung tersumbat | Tidak |

Hasil diagnosis :

konjungtivitis· 86.16%

episkleritis· 8.29%

perdarahan-subkonjungtiva· 2.19%

trikiasis· 2.01%

hordeolum-bintitan· 0.85%

### Uji Coba 6

Laki laki, 67 tahun

|  |  |
| --- | --- |
| **Gejala** | **Jawaban** |
| Kulit ruam | Ya |
| Demam | Ya |
| Demam tinggi | Tidak |
| Demam > 3 hari | Tidak |
| Malaise | Tidak |
| Diare | Tidak |
| Otot nyeri | Tidak |
| Kepala sakit/nyeri | Tidak |
| Batuk | Tidak |
| Tenggorokan sakit | Tidak |
| Penurunan berat badan | Tidak |
| Sendi nyeri | Tidak |
| Kulit bintik merah | Ya |
| Hidung mengeluarkan ingus | Tidak |
| Kulit memerah | Ya |
| Mual | Tidak |
| Muntah | Tidak |
| nafsu makan berkurang | Tidak |
| Sakit perut | Tidak |
| Mengigil | Tidak |
| Sesak nafas | Tidak |

Hasil diagnosis :

morbili· 69.75%

lupus-eritematosus-sistemik· 27.99%

filariasis· 1.01%

lepra· 0.39%

ankilostomiasis· 0.22%

**BIODATA PENULIS**

Kevin Alif Fachreza lahir di Magetan pada 9 Maret 1997 menghabiskan masa kecil di Tangerang Selatan dan masa remaja di Surabaya. Penulis menempuh pendidikan sekolah dasar di SD Tirta Buaran Tangerang (2002 – 2007) dan SD Dewi Sartika Surabaya (2007 - 2008), lalu di SMPN 1 Surabaya (2008 – 2011) terakhir di SMAN 9 Surabaya (2011 – 2014). Setelah menempuh pendidikan SMA penulis melanjutkan kuliah di Teknik Informatika ITS pada 2014 – 2018. Selama menempuh kuliah penulis aktif pada organisasi himpunan jurusan.

Pada 2017 penulis membuat aplikasi Medify yang digunakan oleh banyak dokter dan pada tahun yang sama penulis mendirikan perusahaan PT Global Medika Digitama, dan memiliki jabatan sebagai *Chief Executive Officer*. Penulis dapat dihubungi melalui *email* kevin@medify.id