TUGAS AKHIR - KI000000

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 – KI000000

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 – KI000000

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. 198410162008121002) (Pembimbing 1)

2. Anny Yuniarti, S.Kom, M.Comp.Sc .....
   (NIP.) (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

Penulis ingin mengucapkan terima kasih sebesar-besarnya kepada:

1. Allah SWT.
2. Keluarga penulis
3. Saya Sendiri.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in

reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Surabaya, 07 Juni 2018

Kevin A Fachreza

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR KODE SUMBER

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.

1.1 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.

1.2 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*?

1.3 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.

1.4 Tujuan Pembuatan Tugas Akhir

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

1.5 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.

1.6 Metodologi

Metodologi yang akan digunakan pada tugas akhir ini adalah:

a. 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 menciptakan produk.

b. 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).

c. Analisis dan Desain

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

d. Implementasi

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

e. 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

$$Akurasi : \frac{TP+TN}{TP+TN+FP+FN}$$

TP : True Positive TN : True Negative
FP : False Positive FN : False Negative

f. 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.

1.7 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.

2. 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*.

3. 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*.

4. 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*.

5. 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.

6. 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

BAB II
DASAR TEORI

2.1 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.

2. 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?

3. 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.

4. 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.

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

6. Menanyakan informasi lingkungan keseharian
   Dokter akan menanyakan bagaimana keseharian pasien, apakah pasien merokok, atau apakah pasien menggunakan obat obatan terlarang.

7. 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

8. Mengulas ulang keluhan yang diberikan pasien
   Dokter akan mengulas poin poin penting yang diberikan pasien sebelum memberikan diagnosis.

9. 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 indikasi seberapa sering *item* akan muncul pada dataset. [4]

$$support(A \rightarrow B) = P(A \cup B)$$

2.2.3 Confidence

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

$$confidence(A \rightarrow B) = P(B|A)$$

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 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] :

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

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.

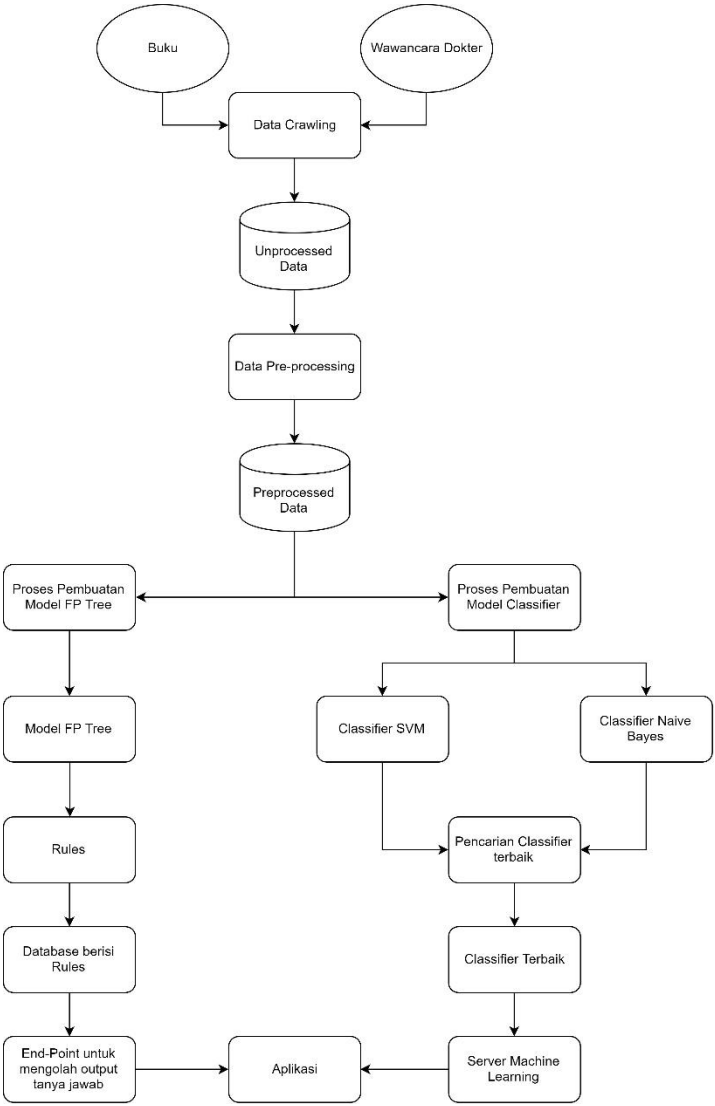
Data ini terdiri dari 858 data kasus, yang memiliki 66 diagnosis atau kelas. Serta 375 atribut. Data ini akan dibagi menjadi 2 kelompok yaitu data belajar dan data pengujian. Data belajar terdiri dari 10 kasus untuk setiap kelas. Dan data pengujian terdiri dari 3 kasus untuk setiap kelas.

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 user. Pada tahap pembuatan model FP Tree, data akan difilter dengan support sesuai dengan skenario. Lalu data support tersebut akan di training untuk membuat rules sesuai. Rules yang dihasilkan oleh FP Tree akan dimasukkan kedalam database yang nantinya akan digunakan oleh server untuk menentukan pertanyaan yang akan ditanyakan kepada pengguna.

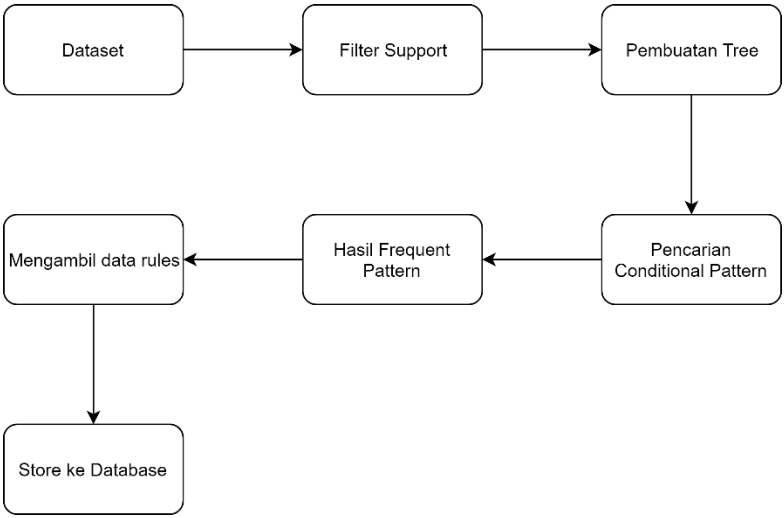
Pada tahap pembuatan model *classifier,* dataset akan di filter berdasarkan support, yang mana akan di training dengan *Naïve Bayes* dan juga *SVM.* Lalu masing masing hasil model akan di ujikan dengan data pengujian. Hasil dari pengujian ini adalah *classifer* dengan hasil yang terbaik, yang mana model 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 3.1 Diagram Sistem Secara Umum

3.2.1 Perancangan Model FP Tree



Gambar 3.2 Diagram Algoritma FP Tree

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

Atribut pada masing masing data akan difilter berdasarkan support. Support yang akan diskenariokan adalah 3, 5, 8, 10, 12, 15, 18, 20, dan 23. Setelah di filter, data akan dibentuk tree berdasarkan urutan support dari yang terbanyak hingga terkecil. Setelah tree terbentuk, maka akan dicari *Frequent Pattern* dari setiap node. Dari *Frequent Pattern* tersebut maka kita dapat mengekstrak informasi rules beserta confidence. 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 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 *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 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 :

$$Akurasi = \frac{TP + TN}{TP + FP + FN + TN}$$

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 :

$$Skor = Peringkat * 20$$

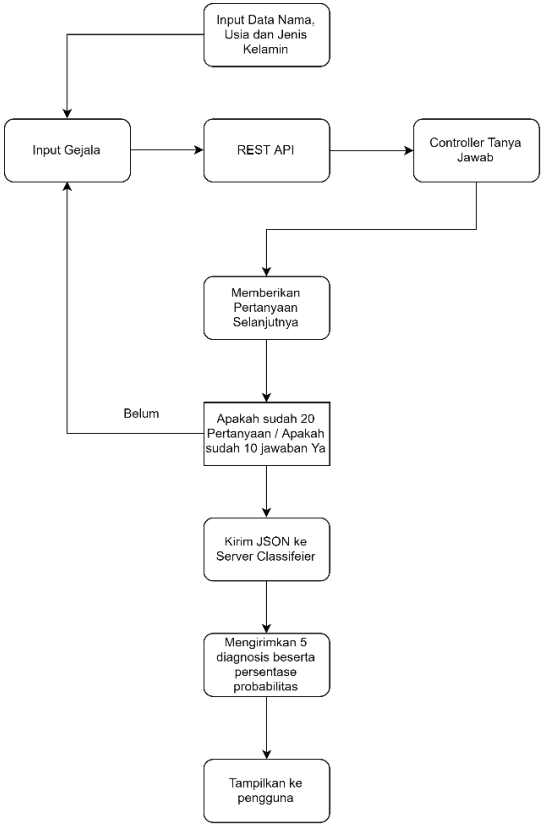
Selain itu juga akan dihitung nilai kemunculan dari hasil diagnosis. Sehingga jika ada salah satu hasil klasifikasi dari 5 hasil yang sesuai dengan hasil pada data training maka akan dinilai benar.

3.2.4 Perancangan Aplikasi

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

a. Perancangan Desain Aplikasi Secara Umm
b. Perancangan Output Tanya Jawab
c. Perancangan Output Diagnosis

3.2.4.1 Perancangan Desain Aplikasi Secara Umum

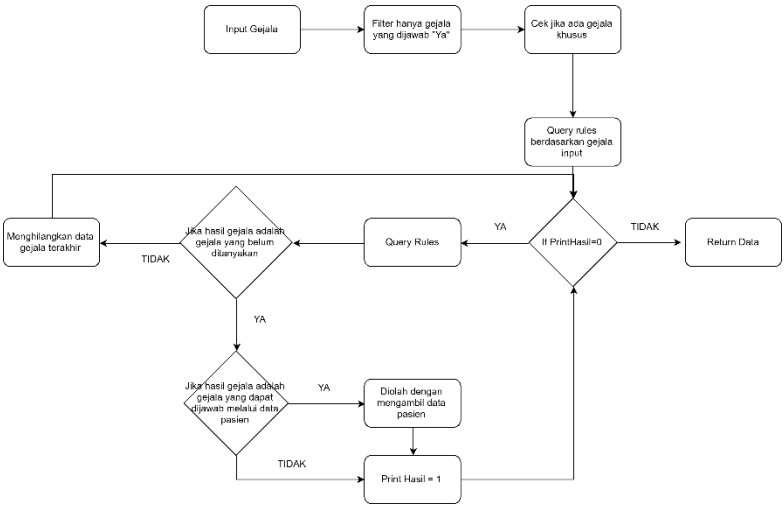


Gambar 3.3 Diagram Alur Aplikasi

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 3.4 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 :

a. Bayi 0 – 1
b. Balita dan Remaja 3 - 18
c. Dewasa 18 – 65
d. 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.

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.

4.1 Lingkungan Implementasi

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

Tabel 4.1 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

4.2 Implementasi

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

4.2.1 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.

4.2.1.1 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.

Kode Sumber 4.1 Filter pada FP Tree

```
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.
6.      for item in temp_item_counter:
7.          if(temp_item_counter[item] <
    min_support):
8.              del item_counter[item]
9.
10. itemOrder = {}
11. lengthItems = len(item_counter)
12.    for i in range(1,lengthItems+1):
```

```
13.         itemOrder[item_counter.most_common()[-
    i][0]] = i;
14.
15. count = 0;
16. for item in items:
17.     item = filter(lambda v: v in itemOrder, item)
18.     item.sort(key=lambda v: itemOrder[v],
    reverse=True)
19.     items[count] = item
20.     count+=1
21. data = [items,item_counter]
22.      return data
```

4.2.1.2 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.

Kode Sumber 4.2 Pembuatan Tree Pada FP Tree

```
1.  def buildTree(dataset):
2.      root = Node("root",support = 0)
3.
4.      for data in dataset:
5.          parent = 0;
6.              for item in data:
7.                  if parent==0:
8.                      parent = root
9.                      child = find(parent,
    filter_=lambda node: node.name == item,
    stop=None, maxlevel = 2)
10.
11.             if child:
12.                 child.support = child.support + 1
```

```
13.                 parent = child
14.             else:
15.                 new_node = Node(item, parent=parent,
    support = 1)
16.                 parent = new_node
17. return root
```

4.2.1.3 Mengekstrak Frequent Pattern

Pada tahap ini dari tree fungsi akan mengekstrak frequent pattern 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.

Kode Sumber 4.3 Ekstrak Frequent Pattern pada FP Tree

```
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.
8.        condPattern = {}
9.        frequentPattern = []
10.
11.       for item in itemReverse:
12.           nodes = findall_by_attr(root, item)
13.           condPatternParents = []
14.           for node in nodes:
15.               parent = str(node)
16.               parent = parent.replace("/root/","")
```

```
17.             parent = parent.split("'")[1]
18.             for i in range(0, node.support):
19.                 condPatternParents.append(parent)
20.         condPattern[item] = condPatternParents
21.
22.     for item in itemReverse:
23.         itemsInPattern = []
24.             for pattern in condPattern[item]:
25.                 temp_pattern = pattern.split('/')
26.                 for temp_item in temp_pattern:
27.                     itemsInPattern.append(temp_item)
28.
29.             item_counter = Counter(itemsInPattern)
30.             temp_item_counter =
    Counter(item_counter)
31.
32.             for temp_item in temp_item_counter:
33.                 if(temp_item_counter[temp_item] <
    min_support):
34.                     del item_counter[temp_item]
35.
36.             willBeCombinatedItem = []
37.             willBeCombinatedItem = filter(lambda v:
    v in item_counter, itemReverse)
38.
39.             for L in range(1,
    len(willBeCombinatedItem)+1):
40.                 for resultcombo in
    combinations(willBeCombinatedItem, L):
41.                     resultcombo = list(resultcombo)
42.                     if item in resultcombo:
43.                         support = 0
44.                         for pattern in
    condPattern[item]:
45.                             pattern_array =
    pattern.split('/')
46.                         flag_support_increment = 1
```

```
47.                         for combo in resultcombo:
48.                             if combo not in
     pattern_array:
49.
     flag_support_increment = 0
50.                             if
     flag_support_increment == 1:
51.                                 support += 1
52.
53.     if support >= min_support:
54.
     frequentPattern.append(FrequentPattern(item,','.
     join(resultcombo) ,str(support)))
55.   return frequentPattern
```

4.2.1.4 Ekstrak Rules

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

Kode Sumber 4.4 Ekstrak Rules pada FP Tree

```
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.
9.                  for item in left:
10.             right.remove(item)
11.             left_string = ','.join(left)
12.             right_string = ','.join(right)
```

```
13.             subset_string =  ','.join(subset)
14.             support_subset =
    find_by_item(filteredList, subset_string)
15.             support_left =
    find_by_item(filteredList, left_string)
16.             support =
    float(float(support_subset)/float(support_left))
17.             right = sorted(right, key=lambda x:
    int(x))
18.             left = sorted(left, key=lambda x:
    int(x))
19.             left_string = ','.join(left)
20.             right_string = ','.join(right)
21.             left_string = left_string.replace("
    ","")
22.             right_string = right_string.replace("
    ","")
23.             query = 'INSERT INTO
    rules_23(items,result,probability)
    VALUES("'+left_string+'","'+right_string+'","'+s
    tr(support)+'");'
24.             cursor.execute(query)
25.                 cnx.commit()
```

4.2.2 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 Bayes
2. Support Vector Machine

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

4.2.2.1 Naïve Bayes

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

Kode Sumber 4.5 Pembuatan Model Naive Bayes

```
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/Naiv
       eBayes/model_architecture.sav'
20.    pickle.dump(model, open(filename, 'wb'))
```

4.2.2.2 Support Vector Machine

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

Kode Sumber 4.6 Pembuatan Model SVM

```
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/Naiv
     eBayes/model_architecture.sav'
19.  pickle.dump(model, open(filename, 'wb'))
```

4.2.3 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.

4.2.3.1 Implementasi Evaluasi Akurasi

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

Kode Sumber 4.7 Evaluasi Akurasi

```
1. def getScore():
2.     filename =
   '../Users/Kevin/PycharmProjects/TugasAkhir/NaiveBa
   yes/model_architecture.sav'
3.     loaded_model = pickle.load(open(filename,
   'rb'))
4.     score = loaded_model.score(X_test, Y_test)
5.     return score
```

4.2.3.2 Implementasi Evaluasi IR

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.

Kode Sumber 4.8 Evaluasi Information Retrieval

```
1.  def getIRScore():
2.      predictions = model.predict_proba(X_test)
3.
4.      index = 0
5.      filewrite =
    '../Users/Kevin/PycharmProjects/TugasAkhir/lapor
    an/t18/svm_prediction_result.txt'
6.      with open(filewrite, 'w') as result_file:
7.          result_file.write('jumlah gejala ' +
    str(jumlah_gejala) + '\n')
8.          result_file.write('jumlah diagnosis ' +
    str(len(class_map)) + '\n')
9.          result_file.write('akurasi ' +
    str(score) + '\n\n')
10.
11.         item_pred_IR = 0;
12.         pred_class_score_array = []
13.         item_count = 0;
14.         for pred in predictions:
15.             top5 = pred.argsort()[-5:][::-1]
16.
17.             item_pred = top5[0]
18.             item_pred_str = class_map[item_pred]
19.
20.             item_true = Y_test[index]
21.
22.             string = str(item_pred_str) + ' ---
    ' + str(item_true)
23.             result_file.write(string+'\n')
24.
25.             pred_class_score = 100;
26.             pred_class_score_fix = 0;
27.             for item in top5:
```

```
28.                 result_file.write(str(class_map[item]) + ' '+
    str(pred[item])+'\n')
29.                 temp_pred = str(class_map[item])
30.                 temp_true = str(item_true)
31.
32.                 if(pred_class_score_fix == 0):
33.                     if(temp_true == temp_pred):
34.                         pred_class_score_fix =
    pred_class_score
35.                         item_pred_IR += 1
36.                     else:
37.                         pred_class_score =
    pred_class_score - 20;
38.
39.
    pred_class_score_array.append(pred_class_score_f
    ix)
40.             result_file.write('skor IR' +
    str(pred_class_score_fix) + '\n')
41.             result_file.write('\n')
```

4.2.4 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

4.2.4.1 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.

Kode Sumber 4.9 Tanya Jawab Pada Server

```
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.        }
15.
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.       }
42.
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' => $rules,
60.                 'append' => 0,
61.             'append_value' => 0
62.             ]);
63.          }
64.       }
65.
66.       foreach($arrayGejalaSpecialUsia as $age)
67.       {
68.          if(in_array($age, $arrayGejalaAsked))
69.          {
70.             array_push($arrayGejalaAsked, 35);
71.             array_push($arrayGejalaAsked, 48);
72.             array_push($arrayGejalaAsked, 331);
73.             array_push($arrayGejalaAsked, 139);
74.             break;
75.          }
76.       }
77.       $arrayGejalaSorted = $arrayGejala;
78.       $tempArrayGejala = $arrayGejala;
79.       sort($arrayGejalaSorted);
80.       $stringGejalaSorted = implode(",",
      $arrayGejalaSorted);
81.       $printResult = 0;
82.       $gejala_data = 0;
83.       $append = 0;
84.       $append_value = 0;
85.       $command = 'ask';
86.       $out = 0;
87.       $found = 0;
88.       $while_iterate_count = 0;
89.       while(!$printResult)
90.       {
```

```
91.          $rules =
       Rules::where('items',$stringGejalaSorted)-
       >orderBy('probability','desc')->get();
92.          if(!empty($rules[0]))
93.          {
94.             if($while_iterate_count > 20)
95.             {
96.                $printResult = 1;
97.                $command = 'out';
98.             }
99.             foreach($rules as $rule)
100.            {
101.               if(!in_array($rule->result,
       $arrayGejalaAsked))
102.               {
103.                  if(in_array($rule->result,
       $arrayGejalaSpecial))
104.                  {
105.                     $gejala_result = explode(',',
       $rule->result);
106.                     $result = $this-
       >getSpecialRule($gejala_result[0],$pasien);
107.                     $command =
       $result['command'];
108.                     $gejala_data =
       $result['gejala'];
109.                     $append = $result['append'];
110.                     $append_value =
       $result['append_value'];
111.                     $printResult = 1;
112.                     $found = 1;
113.                     break;
114.                  }
115.                  else
116.                  {
117.                     $gejala_result = explode(',',
       $rule->result);
```

```
118.                     $gejala_data =
      Gejala::where('id',$gejala_result[0])->first();
119.                     $printResult = 1;
120.                     $command = 'ask';
121.                     $append = 0;
122.                     $found = 1;
123.                     break;
124.                  }
125.               }
126.               else
127.               {
128.                  if(count($tempArrayGejala) > 1)
      //jika tinggal 1 yang belum di pop
129.                  {
130.                     $out = 0;
131.                     array_pop($tempArrayGejala);
132.                     $arrayGejalaSorted =
      $tempArrayGejala;
133.                     sort($arrayGejalaSorted);
134.                     $stringGejalaSorted =
      implode(",", $arrayGejalaSorted);
135.                  }
136.                  elseif(count($tempArrayGejala)
      == 1)
137.                  {
138.                     $out=0;
139.                  }
140.               }
141.            }
142.            if($found !=1 &&
      count($tempArrayGejala) == 1)
143.               $out = 1;
144.            } /*END OF IF RULES[0]*/
145.            else
146.            {
147.               if(count($tempArrayGejala) > 1)
      //jika tinggal 1 yang belum di pop
```

```
148.                {
149.                   $out = 0;
150.                   array_pop($tempArrayGejala);
151.                   $arrayGejalaSorted =
       $tempArrayGejala;
152.                   sort($arrayGejalaSorted);
153.                   $stringGejalaSorted =
       implode(",", $arrayGejalaSorted);
154.                }
155.                elseif(count($tempArrayGejala) ==
       1)
156.                {
157.                   $out=0;
158.                }
159.             }
160.             if($out)
161.             {
162.                if(count($tempArrayGejala) > 1)
       //jika tinggal 1 yang belum di pop
163.                {
164.                   $out = 0;
165.                   array_pop($tempArrayGejala);
166.                   $arrayGejalaSorted =
       $tempArrayGejala;
167.                   sort($arrayGejalaSorted);
168.                   $stringGejalaSorted =
       implode(",", $arrayGejalaSorted);
169.                }
170.                elseif(count($tempArrayGejala) ==
       1)
171.                {
172.                   $out=0;
173.                }
174.             }
175.             $while_iterate_count++;
176.          }
177.       return json_encode([
```

```
178.          'command' => $command,
179.          'gejala' => $gejala_data,
180.          'append' => $append,
181.          'append_value' => $append_value
182.      ]);
183.   }
```

4.2.4.2 Prediksi

Kode Sumber 4.10 Prediksi Pada Server

```
1.   def predict():
2.       total_attributes=376
3.       label =
     pandas.read_csv("../Users/Kevin/PycharmProj
     ects/TugasAkhir/dataset/label.csv")
4.       label = label.values
5.      attributes =
     pandas.read_csv("../Users/Kevin/PycharmProj
     ects/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.
22.     seed = 7
23.     numpy.random.seed(seed)
24.     dataframe =
    pandas.read_csv("../Users/Kevin/PycharmProj
    ects/TugasAkhir/dataset/classifier-
    training-t5.csv", skipinitialspace=True)
25.     dataset = dataframe.values
26.     jumlah_gejala = len(dataset[0]) - 1
27.     X_train = dataset[:,0:jumlah_gejala]
28.     Y_train = dataset[:,jumlah_gejala]
29.  #assigning testing
30.     dataframe =
    pandas.read_csv("../Users/Kevin/PycharmProj
    ects/TugasAkhir/dataset/classifier-testing-
    t5.csv", skipinitialspace=True)
31.     dataset = dataframe.values
32.       X_test = dataset[:,0:jumlah_gejala]
33.  Y_test = dataset[:,jumlah_gejala]
34.  #label
35.     label =
    pandas.read_csv("../Users/Kevin/PycharmProj
    ects/TugasAkhir/dataset/label.csv")
36.     label = label.values
37.  #Create a Gaussian Classifier
```

```
38.    model = BernoulliNB()
39.  # Train the model using the training sets
40.    y_pred = model.fit(X_train,
     Y_train).predict(X_test)
41.    class_map = model.classes_
42.    score = model.score(X_test, Y_test)
43.    predictions =
     model.predict_proba(gejala_array_np)
44.    result = []
45.    jsondata = '{ "result":['
46.    index = 0
47.    for pred in predictions:
48.      top5 = pred.argsort()[-5:][::-1]
49.        for item in top5:
50.          labels = label[item][0]
51.
     result.append(HasilDiagnosis(labels,pred[it
     em]))
52.          json_item_string = '{"diagnosis":"'
     + str(class_map[item])
     +'","probability":"'+ str(pred[item]) +'"}'
53.          if index < 4:
54.            json_item_string =
     json_item_string + ','
55.         index = index+1
56.         jsondata = jsondata +
     json_item_string
57.    jsondata = jsondata + "]}"
58.    format_jsondata = json.dumps(jsondata)
59.    return jsondata
```

4.2.4.3 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.

Kode Sumber 4.11 Interface HTML Pada User

```
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 4.12 Javascript Tanya Jawab Pada Interface 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 4.13 Javascript Diagnosis Pada 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
```

BAB V
PENGUJIAN DAN EVALUASI

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

5.1 Lingkungan Pengujian

Lingkungan pengujian yang digunakan adalah :

Tabel 5.1 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 • Google Chrome

5.2 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.

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

5.3 Skenario Uji Coba

Skenario yang dijalankan pada evaluasi ini berupa perhitungan akurasi dan information retrieval. Pengujian akan dimulai dengan filter support untuk gejala gejala. Ada 9 jumlah 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 text.

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.

5.4 Hasil Uji Coba

Pada sub bab ini akan dijelaskan bagaimana hasil dari hasil uji coba dari evaluasi secara sistem dan juga evaluasi oleh dokter.

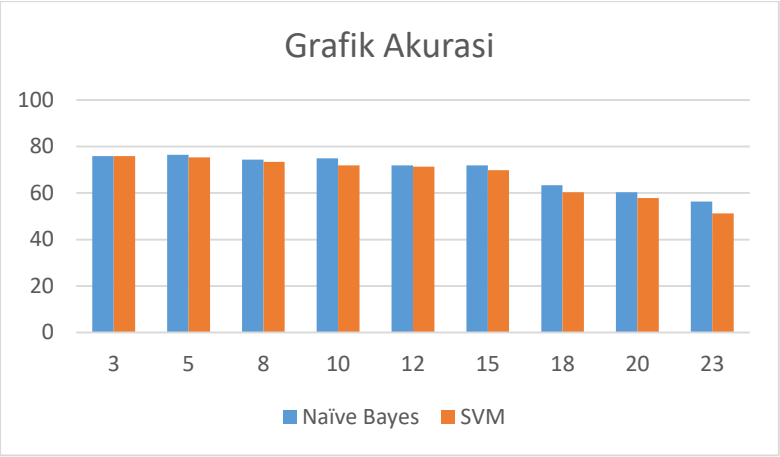
5.4.1 Hasil Uji Skenario Sistem

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

5.4.1.1 Akurasi

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

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

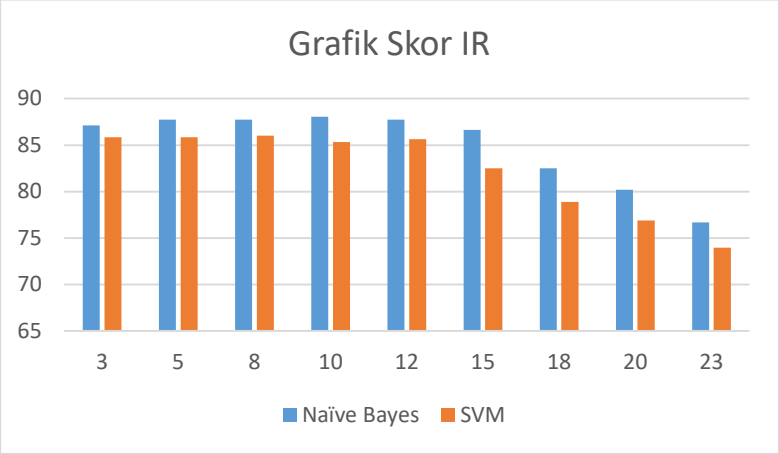


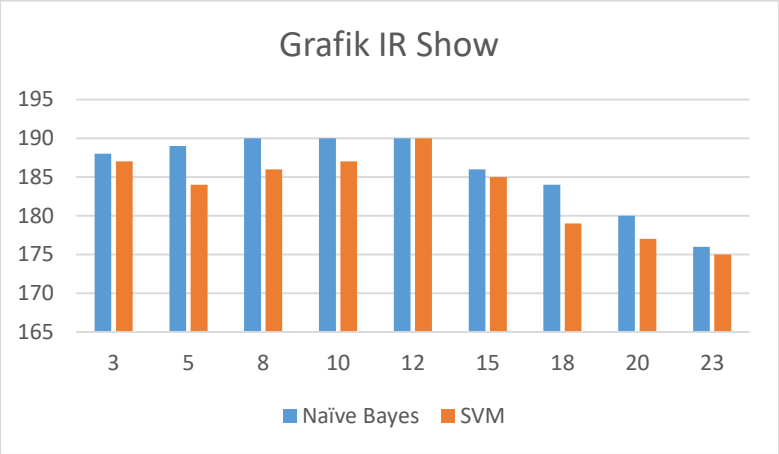
Pada hasil akurasi dapat dilihat bahwa grafik cenderung menurun walaupun dengan selisih yang sangat kecil. Pada saat support bernilai 5, Naïve Bayes memiliki akurasi tertinggi sedangkan SVM memiliki akurasi tertinggi pada saat support bernilai 3.

5.4.1.2 Information Retrieval

Threshold	Naïve Bayes		SVM	
	IR Score	IR Show	IR Score	IR Show
3	87.135	188	85.829	187
5	87.738	189	85.829	184
8	87.738	190	86.030	186
10	88.040	190	85.326	187
12	87.738	190	85.628	190
15	86.633	186	82.512	185
18	82.512	184	78.894	179

20	80.201	180	76.884	177
23	76.683	176	73.969	175





Pada evaluasi ini dapat dilihat grafik IR Score cenderung stabil sedangkan IR Show cenderung fluktuatif. Dengan IR Show

dan IR Score disaat support 3 – 12 selisih antar hasil tidak terlalu banyak. Tetapi pada saat support bernilai 15 kedua classifier mengalami penurunan yang cukup signifikan.

5.4.2 Hasil Uji Pengguna

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%

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.

6.1 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 support bernilai 5 dengan akurasi 76,381%, skor IR 87,738% dan IR Show 189. Hal ini membuktikan support berpengaruh pada performa classifier.

6.2 Saran

Saran yang diberikan untuk pengembangan aplikasi ini kedepannya adalah :

1. 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.

2. 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].

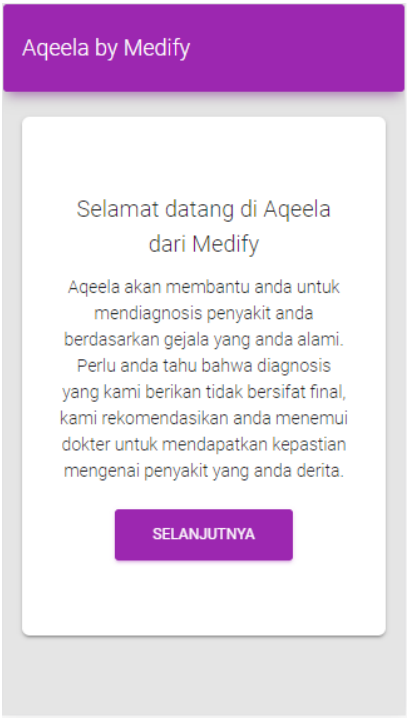
[12] A. Ronacher, "Flask," Flask, 2013. [Online]. Available: https://web.archive.org/web/20171117015927/http://flask.pocoo.org:80/docs/0.10/foreword. [Accessed 30 May 2018].

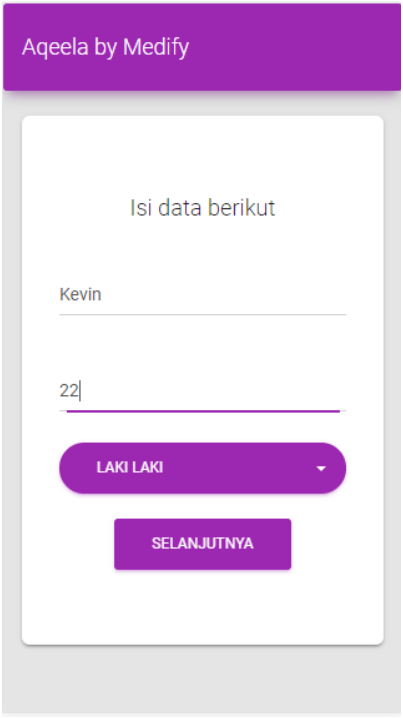
[13] Laravel, "Github," Github Inc, 19 May 2018. [Online]. Available: https://github.com/laravel/framework. [Accessed 30 May 2018].

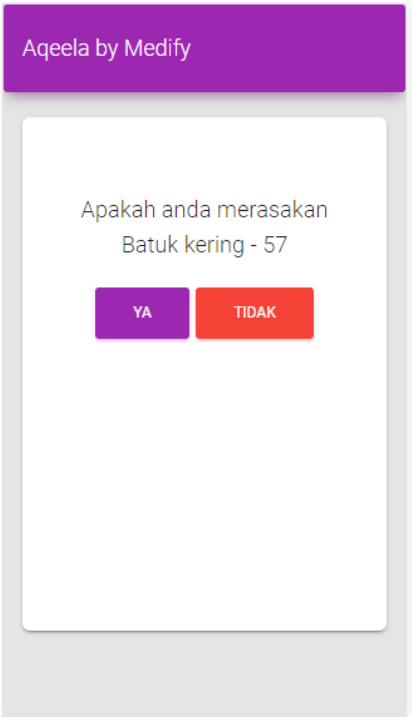
[14] J. P. a. Y. Y. Jiawei Han, "Mining Frequent Patterns without Candidate Generation".

LAMPIRAN

1. Tampilan aplikasi






Aqeela by Medify
Apakah anda merasakan
Batuk kering - 57
YA
TIDAK


Aqeela by Medify
Penyakit yang mungkin kamu
derita
faringitis 92.24%
influenza 3.00%
rhinitis 2.14%
leptospirosis 0.83%
ankilostomiasis 0.41%