Perbandingan Kinerja Metode Problem Transformation-KNN dan Algorithm Adaptation-KNN pada Klasifikasi Multi-Label Pertanyaan Kotakode

Erlina Eka Fitriani[1], Wiyli Yustanti[2]

[1,2] Sistem Informasi, Teknik Informatika, Universitas Negeri Surabaya
[1]erlinaeka.18009@mhs.unesa.ac.id
[3]wiyliyustanti@unesa.ac.id

Abstrak*— Klasifikasi multi-label merupakan proses pengelompokan data ke dalam beberapa kelas berdasarkan kesamaan ciri atau karakteristik data. Pada klasifikasi multi-label setiap data dapat memiliki lebih dari satu kelas. Implementasi klasifikasi multi-label dapat dilakukan melalui dua metode pendekatan yaitu *Problem Transformation* dan *Algorithm Adaptation*. Penelitian topik klasifikasi multi-label teks telah dilakukan oleh para peneliti terdahulu. Akan tetapi, belum terdapat penelitian yang berfokus pada perbandingan *Problem Transformation* dan *Algorithm Adaptation* berdasarkan pertanyaan multi-label dengan tagar pertanyaan sebagai label atau kelas. Tujuan penelitian ini adalah melakukan klasifikasi multi-label pada data teks dan membandingkan hasil kinerja metode *Problem Transformation* dan *Algorithm Adaptation* dalam melakukan klasifikasi multi-label. Dataset yang digunakan adalah 450 data pertanyaan pada forum Q&A *platform* Kotakode. Metode *Problem Transformation* yang digunakan pada penelitian ini adalah *Label Powerset, Binary Relevance,* dan *Classifier Chain* dengan *K-Nearest Neighbor* sebagai algoritma klasifikasi. Sedangkan metode *Algorithm Adaptation* yang digunakan adalah Multi-Label *K-Nearest Neighbor* (ML-KNN). *Grid Search Cross Validation* digunakan pada penelitian ini untuk menemukan nilai *hyperparameter* k yang dapat memberikan hasil kinerja model terbaik. Hasil Penelitian menunjukkan bahwa metode *Problem Transformation Label Powerset* dengan *K-Nearest Neighbor* sebagai algoritma klasifikasi menghasilkan nilai akurasi, *precision, recall,* dan *f1 score* terbaik, yaitu 86%, 92%, 86%, dan 87%. Berdasarkan hasil tersebut, Metode *Problem Transformation Label Powerset*-KNN menghasilkan kinerja lebih baik dalam melakukan klasifikasi multi-label pertanyaan Kotakode dibandingkan dengan metode *Algorithm Adaptation* Multi-Label *K-Nearest Neighbor.

***Kata Kunci*— Klasifikasi Multi-Label, *Problem Transformation*, ML-KNN, *Grid Search Cross Validation*.**

I. PENDAHULUAN

Kotakode merupakan *platform* komunitas para pegiat bidang IT di Indonesia untuk belajar dan berbagi wawasan seputar bidang IT. Setiap pengguna *platform* Kotakode dapat mengajukan pertanyaan terkait pemrograman melalui sebuah forum Q&A yang disediakan. Setiap pertanyaan yang diajukan harus memiliki tagar untuk mengelompokkan pertanyaan sesuai dengan tagar dan memudahkan pengguna lain dalam memberikan solusi atau jawaban. Pemberian tagar pada pertanyaan yang tidak sesuai ataupun kurang dapat terjadi ketika pengguna memberikan tagar pertanyaan. Metode klasifikasi berdasarkan pertanyaan yang diajukan pengguna, dinilai lebih efektif dan efisien apabila dibandingkan dengan pemberian tagar berdasarkan perkiraan pengguna. Metode klasifikasi dinilai lebih efektif dan efisien karena tidak mengharuskan pengguna untuk memperkirakan tagar yang sesuai dengan pertanyaan sehingga jawaban ataupun tanggapan yang dihasilkan lebih relevan dan akurat [1]. Metode klasifikasi yang digunakan adalah klasifikasi multi-label. Klasifikasi multi-label sesuai untuk menyelesaikan permasalahan tersebut karena setiap pertanyaan dapat memiliki lebih dari satu tagar yang digunakan sebagai label atau kelas [2]. Implementasi klasifikasi multi-label dapat dilakukan melalui dua metode pendekatan yaitu *Problem Transformation* dan *Algorithm Adaptation* [3].

Penelitian topik klasifikasi multi-label telah dilakukan oleh para peneliti terdahulu. Akan tetapi belum terdapat penelitian yang berfokus pada perbandingan kinerja antara metode *Problem Transformation* dan *Algorithm Adaptation* berdasarkan pertanyaan multi-label dengan tagar pertanyaan sebagai label atau kelas. Salah satu penelitian yang melakukan klasifikasi pertanyaan adalah klasifikasi pertanyaan bidang akademik [4]. Klasifikasi yang dilakukan pada penelitian tersebut adalah klasifikasi single-label dengan 5W 1H sebagai label atau kelas dan menggunakan algoritma klasifikasi *K-Nearest Neighbor*. Penelitian tentang klasifikasi multi-label pernah dilakukan oleh Muhammad Okky Ibrohim dan Indra Budi pada tahun 2019. Penelitian tersebut melakukan klasifikasi multi-label ujaran kebencian pada *twitter* dan membandingkan *Label Powerset, Binary Revelance,* dan *Classifier Chain* sebagai *Problem Transformation* [5]. Akan tetapi, pada penelitian tersebut tidak dilakukan perbandingan kinerja metode *Problem Transformation* dengan *Algorithm Adaptation*. Begitu pula pada penelitian klasifikasi topik multi-label pada Hadis Bukhari dalam terjemahan Bahasa Indonesia [6]. Penelitian tersebut membandingkan metode *Problem Transformation Label Powerset* dengan *Binary Relevance*.

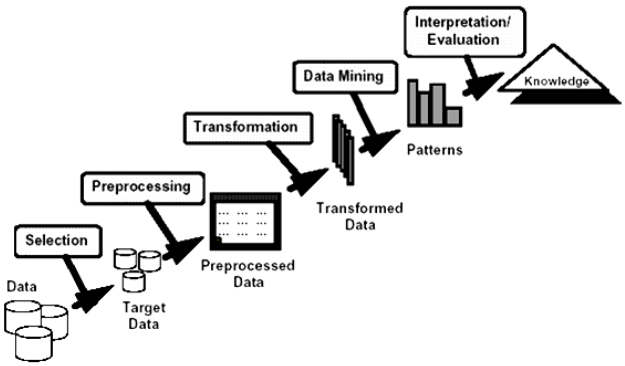
Metode *Problem Transformation* dan *Algorithm Adaptation* memiliki keunggulan masing-masing dalam melakukan klasifikasi multi-label. *Label Powerset* sebagai metode *Problem Transformation* memiliki keunggulan dapat mempertimbangkan hubungan antar label atau kelas. *Binary Relevance* sebagai metode *Problem Transformation* memiliki keunggulan konseptual sederhana serta proses komputasi relatif cepat dan efisien. *Classifer Chain* sebagai metode *Problem Transformation* memiliki keunggulan dapat memberikan peningkatan kinerja prediktif [7]. *K-Nearest*

Neighbor (KNN) sebagai algoritma klasifikasi memiliki keunggulan sederhana dan efektif untuk melakukan klasifikasi teks [8]. Proses komputasi ringan untuk data yang tidak kompleks juga menjadi salah satu keunggulan algoritma KNN [4]. Sedangkan ML-KNN sebagai *Algorithm Adaptation* memiliki keunggulan dapat mengurangi *hamming loss* sehingga dapat meningkatkan kinerja dibandingkan dengan *Algoritma Adaptation* yang lain [7].

Berdasarkan uraian di atas, penelitian ini akan berfokus pada klasifikasi multi-label pertanyaan Kotakode dan membandingkan hasil kinerja metode *Problem Transformation*-KNN dan *Algorithm Adaptation* ML-KNN dalam melakukan klasifikasi multi-label. Penelitian ini bertujuan untuk melakukan klasifikasi multi-label menggunakan metode *Problem Transformation* dengan algoritma klasifikasi *K-Nearest Neighbor* (KNN) dan metode *Algorithm Adaptation Multi-Label K-Nearest Neighbor* (ML-KNN) serta membandingkan hasil kinerja setiap metode untuk mengetahui metode terbaik dalam melakukan klasifikasi multi-label pertanyaan Kotakode. *Term Frequency* and *Inverse Document Frequency* (TF-IDF) digunakan sebagai pembobotan fitur dan *Grid Search Cross Validation* untuk menemukan nilai *hyperparameter* k yang dapat memberikan hasil kinerja model terbaik dalam klasifikasi multi-label pertanyaan Kotakode. Sedangkan, metode *Problem Transformation* yang digunakan adalah *Label Powerset*, *Binary Relevance* dan *Classifier Chain.*

II. METODE PENELITIAN

Pada penelitian ini tahapan klasifikasi multi-label dilakukan mengikuti alur proses *Knowledge Discovery in Database* (KDD). *Knowledge Discovery in Database* merupakan suatu proses terorganisir untuk mengidentifikasikan pola dari dataset yang besar dan kompleks sehingga pola dari data tersebut dapat dipahami dan digunakan. *Data mining* merupakan salah satu tahapan dari serangkaian proses KDD [9]. Gambaran umum tahapan pada proses KDD dapat dilihat pada Gbr 1.



Gbr. 1 Tahapan proses *Knowledge Discovery in Database* (KDD)

A. Data Selection

Data selection merupakan tahap pengumpulan dan pemilihan data yang akan digunakan pada proses *data mining*. Dataset yang digunakan adalah data pertanyaan pada forum Q&A *platform* Kotakode. Data pertanyaan yang dikumpulkan adalah pertanyaan dengan tagar lebih dari satu dan diajukan oleh pengguna *platform* Kotakode tidak lebih dari tanggal 28 Apil 2022. Batasan multi-label yang digunakan pada dataset pertanyaan tersebut adalah 3 label untuk setiap pertanyaan. 17 tagar akan ditetapkan sebagai label, yaitu *css, html, bootstrap, javascript, node.js, react.js, python, machine learning, php, laravel, codeigniter, mysql, kotlin, android, flask, dart*, dan *flutter.*

B. Data Preprocessing

Data Preprocessing dilakukan pada dataset sebelum proses klasifikasi. Tahap *data preprocessing* dilakukan karena dataset memungkinkan memiliki *noise* yang dapat mempengaruhi proses klasifikasi. Data *preprocessing* pada penelitian ini disebut sebagai *text preprocessing* karena dataset yang digunakan berupa data teks. Berikut adalah tahapan *text preprocessing* pada penelitian ini:

1. *Cleaning:* Proses membersihkan data pertanyaan dari *tag html*, tanda baca, spasi berlebih, nomor, dan *link url.*
2. *Case Folding:* Proses yang dilakukan untuk mengubah teks pertanyaan menjadi *lowercase* atau huruf kecil.
3. *Tokenizing:* Proses pemenggalan teks berdasarkan setiap kata atau frasa.
4. *Normalization:* Proses pengubahan penulisan kata yang tidak lengkap, kesalahan dalam penulisan atau *typo* kedalam kata normal sesuai dengan Kamus Besar Bahasa Indonesa (KBBI).
5. *Stemming:* Proses yang dilakukan untuk mengubah setiap kata ke bentuk dasar.
6. *Stopword Removal:* Proses penghapusan kata yang termasuk kedalam *stoplist* (kata tidak penting).

C. Data Transformation

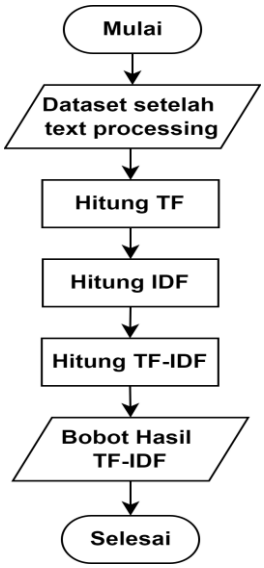
Pada tahap *data transformation* dilakukan pembobotan fitur sehingga akan terbentuk matriks baru yang merepresentasikan bobot dari setiap fitur. Pada penelitian ini *Term Frequency-Invers Document Frequency* (TF-IDF) digunakan untuk melakukan pembobotan fitur. TF-IDF merupakan statistik numerik untuk menggambarkan seberapa penting suatu *term* atau kata dalam suatu dokumen [10].

Gbr 2 menunjukkan alur proses TF-IDF yang dilakukan. Berikut adalah penjelasan setiap tahapan pada proses TF-IDF:

1. Menghitung nilai TF

   Nilai TF menyatakan jumlah kemunculan suatu *term* atau kata pada setiap dokumen. Semakin besar nilai TF maka semakin sering *term* atau kata tersebut muncul pada dokumen. Rumus menghitung nilai TF dapat dilihat pada persamaan (1).

$$TF = F_{t,d} \quad (1)$$



Gbr. 2 Alur proses TF-IDF

2. Menghitung Nilai IDF

Nilai IDF adalah *invers* jumlah dokumen yang memuat suatu *term* atau kata. Rumus menghitung nilai IDF dapat dilihat pada persamaan (2).

$$IDF = \log\left(\frac{N}{dft}\right) \quad (2)$$

3. Menghitung TF-IDF

perhitungan untuk melakukan pembobotan menggunakan TF-IDF adalah dengan cara melakukan perkalian nilai TF dengan IDF. Rumus menghitung nilai IDF dapat dilihat pada persamaan (3).

$$W_{t,d} = TF \times IDF$$
$$W_{t,d} = F_{t,d} \times \log\left(\frac{N}{dft}\right) \quad (3)$$

Keterangan :
$F_{t,d}$ = jumlah *term* (t) pada dokumen (d)
N = jumlah keseluruhan dokumen dalam dataset
d_{ft} = jumlah dokumen yang mengandung *term* (t)

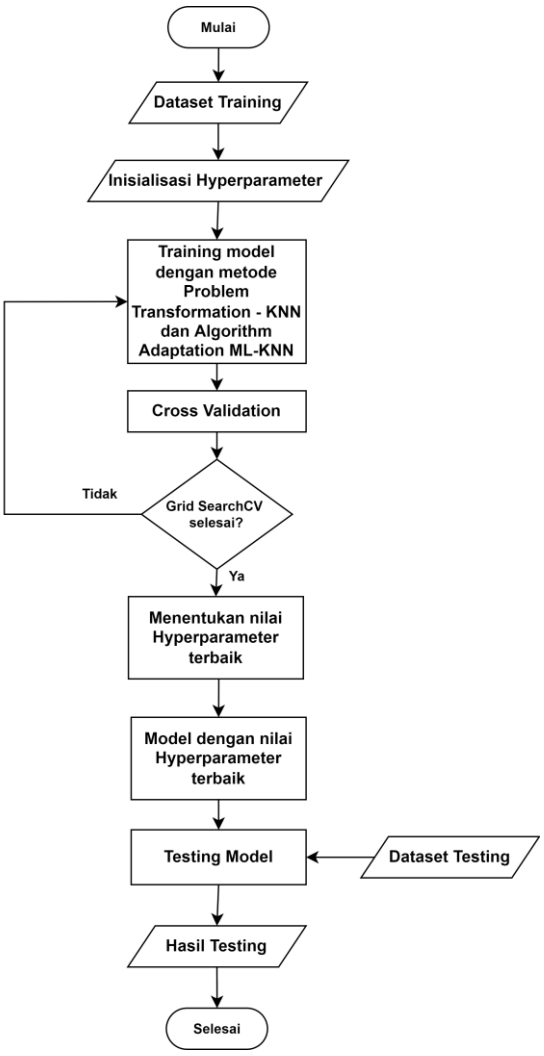
D. Data Mining

Proses data mining dilakukan untuk menemukan pola pada dataset. Metode *Data Mining* yang digunakan harus sesuai dengan tujuan penelitian [9]. Pada penelitian ini metode *data mining* yang digunakan adalah klasifikasi multi-label. Data *training* atau data latih ditentukan sejumlah 80% dari keseluruhan data. Sedangkan dataset *testing* atau data uji ditentukan sejumlah 20% dari keseluruhan data.

Pada proses klasifikasi multi-label dilakukan *hyperparameter tunning* menggunakan *Grid SearchCV*. *Hyperparameter tuning* dilakukan dengan tujuan untuk mendapatkan nilai *hyperparamete*r yang memberikan hasil kinerja model klasifikasi multi-label terbaik. Nilai *hyperparamete*r yang dilakukan pencarian menggunakan *Grid SearchCV* adalah nilai *hyperparameter* k. Rentang nilai k pada proses *hyperparameter tuning* oleh *Grid SeachCV* adalah 1 sampai hasil pencarian batas maksimal nilai k. Batas maksimal nilai k dapat ditentukan berdasarkan hasil akar kuadrat dari jumlah data *testing* [11].

$$K = \sqrt{N} \quad (4)$$

Persamaan 4 merupakan rumus batas maksimal nilai k. N merupakan jumlah data *testing*. *Grid SearchCV* dilakukan terhadap *data training* setelah ditentukan rentang nilai *hyperparameter* k. Nilai *hyperparameter cross validation* yang digunakan pada proses *Grid SearchCV* adalah 3, 5 dan 10. Nilai 3, 5 dan 10 digunakan sebagai parameter *cross validation* dikarenakan nilai tersebut dapat menghasilkan kinerja model yang efektif [12]. Setelah *Grid Search CV* menentukan nilai *hyperparameter* terbaik, dilakukan *testing* atau pengujian terhadap model dengan nilai *hyperparameter* terbaik dari setiap metode klasifikasi multi-label dalam melakukan prediksi data *testing*.



Gbr. 3 Alur proses klasifikasi multi-label

Gbr 3 menunjukan alur proses *data mining* yang dilakukan pada penelitian ini. Metode klasifikasi multi-label yang digunakan pada penelitian ini adalah *Problem Transformation*-KNN dan *Algorithm Adaptation* ML-KNN.

Problem Transformation merupakan metode yang mengubah permasalahan klasifikasi multi-label ke dalam satu atau lebih klasifikasi *single*-label. Pada penelitian ini metode *Problem Transformation* yang digunakan adalah *Label Powerset, Binary Relevance*, dan *Classifier Chain.*

pertanyaan	html	css	bootstrap
pertanyaan 1	1	1	0
pertanyaan 2	0	1	1
pertanyaan 3	1	0	1
pertanyaan 4	1	1	1
pertanyaan 5	1	1	0

pertanyaan	Tagar
pertanyaan 1	110
pertanyaan 2	011
pertanyaan 3	101
pertanyaan 4	111
pertanyaan 5	110

keterangan :
Input
Output

Gbr. 4 Ilustrasi proses *Problem Transformation Label Powerset*

Gbr 4 menunjukkan cara kerja metode *Label Powerset.* Label Powerset bekerja dengan cara mengubah dataset multi-label menjadi dataset multi-*class.* Sehingga ketika dilakukan *Problem Transformation* dengan *Label Powerset*, setiap kombinasi label yang berbeda merupakan satu label baru [13].

pertanyaan	html	css	bootstrap
pertanyaan 1	1	1	0
pertanyaan 2	0	1	1
pertanyaan 3	1	0	1
pertanyaan 4	1	1	1
pertanyaan 5	1	1	0

pertanyaan	html
pertanyaan 1	1
pertanyaan 2	0
pertanyaan 3	1
pertanyaan 4	1
pertanyaan 5	1

pertanyaan	css
pertanyaan 1	1
pertanyaan 2	1
pertanyaan 3	0
pertanyaan 4	1
pertanyaan 5	1

pertanyaan	bootstrap
pertanyaan 1	0
pertanyaan 2	1
pertanyaan 3	1
pertanyaan 4	1
pertanyaan 5	0

keterangan :
Input
Output

Gbr. 5 Ilustrasi proses *Problem Transformation Binary Relevance*

Gbr 5 menunjukkan cara kerja metode *Binary Relevance. Binary Relevance* bekerja dengan cara mengubah dataset *multi-label* menjadi dataset *single*-label sebanyak jumlah label pada dataset [13].

pertanyaan	html	css	bootstrap
pertanyaan 1	1	1	0
pertanyaan 2	0	1	1
pertanyaan 3	1	0	1
pertanyaan 4	1	1	1
pertanyaan 5	1	1	0

pertanyaan	html
pertanyaan 1	1
pertanyaan 2	0
pertanyaan 3	1
pertanyaan 4	1
pertanyaan 5	1

pertanyaan	html	css
pertanyaan 1	1	1
pertanyaan 2	0	1
pertanyaan 3	1	0
pertanyaan 4	1	1
pertanyaan 5	1	1

pertanyaan	html	css	bootstrap
pertanyaan 1	1	1	0
pertanyaan 2	0	1	1
pertanyaan 3	1	0	1
pertanyaan 4	1	1	1
pertanyaan 5	1	1	0

keterangan :
Input
Output

Gbr. 6 Ilustrasi proses *Problem Transformation Classifier Chain*

Gbr 6 menunjukkan cara kerja metode *Classifier Chain. Classifier Chain* melakukan *training* pada model sejumlah k atau sebanyak jumlah label yang ada. Proses *training* pertama dilakukan hanya menggunakan atribut *input* yang asli. Hasil dari proses pertama akan ditambahkan sebagai atribut *input* untuk melakukan proses *training* berikutnya [13].

Setelah *Problem Transformation* dilakukan untuk transformasi dataset multi-label ke dataset *single*-label, algoritma klasifikasi KKN dilakukan untuk klasifikasi dataset pertanyaan Kotakode. Berikut adalah tahapan KNN :

1. Menentukan parameter k, k merupakan tetangga terdekat.
2. Menghitung jarak antara data baru dengan semua data yang ada pada data training menggunakan rumus perhitungan *euclidean distance*.

   Berikut adalah rumus persamaan *euclidean distance* :

   $$dist(X_1, X_2) = \sqrt{\sum_{i=1}^{n}(x_{1i} - x_{2i})^2} \qquad (5)$$

   Keterangan :
   n = dimensi data
   X_1 = data *testing*
   X_2 = data *training*
3. Mengurutkan hasil perhitungan jarak.
4. Menentukan tetangga terdekat berdasarkan jarak terdekat dan nilai parameter k.
5. Menghitung hasil *voting* dari mayoritas k tetangga terdekat sebagai prediksi label atau kelas data baru.

Selain menggunakan metode pendekatan *Problem Transformation, Algorithm Adaptation* digunakan sebagai metode klasifikasi multi-label pertanyaan Kotakode. *Algorithm Adaptation* merupakan metode yang memperluas pembelajaran algoritma tertentu untuk dapat menangani data multi-label secara langsung. Multi-Label KNN digunakan sebagai metode *Algorithm Adaptation* pada penelitian ini. ML-KNN menggunakan prinsip *maximum a posteriori* untuk menentukan set label dari setiap data baru, berdasarkan *prior* dan *posterior probability* dari frekuensi setiap label dalam k tetangga terdekat [13].

E. Evaluation

Proses evaluasi dilakukan pada setiap metode untuk mengetahui kinerja metode dalam klasifikasi multi-label pertanyaan Kotakode. Pada penelitian ini, *confusion matrix* digunakan untuk melakukan proses evaluasi sehingga dapat diketahui nilai *precision, recall,* dan *f1 score*.

TABEL I
CONFUSION MATRIX

Actual Class	*Predicted Class*		
		+	-
	+	TP	FN
	-	FP	TN

Keterangan :

True Positive (TP) = jumlah data berlabel positif yang diprediksi secara tepat.

True Negative (TN) = jumlah data berlabel negatif yang diprediksi secara tepat.

False Positive (FP) = jumlah data berlabel negatif yang diprediksi sebagai label positif.

False Negative (FN) = jumlah data berlabel positif yang diprediksi sebagai label negatif.

Berdasarkan *confusion matrix* pada tabel 1, nilai hasil evaluasi metode dapat dihitung menggunakan persamaan berikut :

Akurasi adalah presentase dari data yang diklasifikasi secara tepat oleh model. Akurasi dihitung menggunakan persamaan 6.

$$Akurasi = \frac{TP+TN}{TP+FP+FN+TN} \quad (6)$$

Precision adalah prediksi benar positif dibandingkan dengan keseluruhan data yang diprediksi positif [14]. *Precision* dihitung menggunakan persamaan 7.

$$Precision = \frac{TP}{TP+FP} \quad (7)$$

Recall adalah prediksi benar positif dibandingkan dengan keseluruhan data yang sebenarnya berlabel positif [14]. *Recall* dihitung menggunakan persamaan 8.

$$Recall = \frac{TP}{TP+FN} \quad (8)$$

f1 score adalah *harmonic mean* dari nilai *precision* dan *recall* [14]. *f1 score* dihitung menggunakan persamaan 9.

$$F1 - score = 2 \times \frac{Precission \times Recall}{Precission + Recall} \quad (9)$$

III. HASIL DAN PEMBAHASAN

Pada tahap ini, dilakukan pembahasan hasil dari pengolahan data berdasarkan metode penelitian yang dijelaskan pada bagian sebelumnya. Pengolahan data dilakukan dengan bahasa *python* menggunakan *software jupyter notebook*. *Jupyter notebook* merupakan lingkungan komputasi interaktif berbasis web untuk membuat dokumen *notebook Jupyter*. *Jupyter* mendukung lebih dari 40 bahasa pemrograman termasuk *python*, R, Julia, dan *Scala*. *Jupyter notebook* diakses melalui *web browser*. Hal tersebut membuat *jupyter notebook* mudah dan praktis untuk digunakan karena satu satunya perangkat lunak yang dibutuhkan oleh 23 pengguna secara lokal hanya *web browser* [15].

A. Data Selection

Pengumpulan data dilakukan dengan cara *srapping link* pertanyaan Kotakode menggunakan *software Octoparse*. *Octoparse* adalah *software web scraping* yang dapat digunakan untuk mengekstrak dan melakukan pengambilan data dari sebuah *web*[16].

Data pertanyaan yang dikumpulkan adalah data pertanyaan multi-label dengan batasan 3 label dan diajukan oleh pengguna pada *platform* Q&A Kotakode tidak lebih dari tanggal 28 April 2022. Pertanyaan yang dikumpulkan yaitu sejumlah 450 pertanyaan. Proses *scrapping link* pertanyaan menghasilkan dataset pertanyaan dalam format *excel* dengan *field* headingPertanyaan, bodyPertanyaan dan tagar pertanyaan. *Field* headingPertanyaan dan bodyPertanyaan digabung menjadi satu *field* yaitu *field* pertanyaan. Tabel II menunjukan rincian jumlah data berdasarkan tagar data pertanyaan multi-label.

TABEL II
RINCIAN JUMLAH DATA

Tagar Pertanyaan	Jumlah
html dan css	36
html, css dan bootstrap	48
javascript dan node.js	38
javascript dan reactjs	51
android dan kotlin	36
python dan machine-learning	41
flutter dan dart	26
php dan codeigniter	48
php dan laravel	51
php dan mysql	49
python dan flask	26
Total Data	**450**

B. Data Preprocessing

Preprocessing dilakukan pada dataset sebelum proses klasifikasi multi-label. Pada tahap *preprocessing* dilakukan beberapa proses yaitu *cleaning data, case folding, tokenizing, normalization, stemming,* dan *stopword removal*. Berikut adalah contoh data sebelum dan sesudah proses *preprocessing* :

TABEL III
CONTOH DATA SEBELUM PREPROCESSING

Sebelum Preprocessing
Mengapa format Top-margin CSS tidak dapat bekerja? Saat saya memasukkan value margin pada sebuah div dalam div lainnya, value top tampaknya diabaikan. Mengapa ini terjadi? #outer { width: 500px; height: 200px; background: #FFCCCC; margin: 50px auto 0 auto; display: block; } #inner { background: #FFCC33; margin: 50px 50px 50px 50px; padding: 10px; display: block; } Hello world!

TABEL IV
CONTOH DATA SESUDAH PREPROCESSING

Setelah Preprocessing
'format', 'top', 'margin', 'css', 'masuk', 'value', 'margin', 'div', 'div', 'value', 'top', 'width', 'height', 'background', 'margin', 'auto', 'auto', 'display', 'block', 'background', 'margin', 'padding', 'display', 'block', 'world'

C. *Data Transformation*

Pada tahap data *transformation* dilakukan pembobotan fitur pada setiap *term* atau kata menggunakan TF-IDF. Langkah TF-IDF dapat dilihat pada Gbr 2. Berikut adalah contoh hasil proses TF-IDF :

TABEL V
CONTOH HASIL PEMBOBOTAN TF-IDF

Term	Data			
	D1	D2	D3	D4
bootstrap	0,190676	0,398832	0,099839	0,566427
button	0	0	0,061941	0
justify	0,57133	0	0	0
warna	0	0,109047	0	0
bentuk	0,130335	0	0	0
font	0	0,174865	0	0

D. *Data Mining*

Pada tahap *data mining* dilakukan klasifikasi multi-label menggunakan metode *Problem Transformation*-KNN dan *Algorithm Adaptation* ML-KNN. *Grid SearchCV d*igunakan pada proses klasifikasi multi-label sebagai *tunning hyperparameter*. *Grid SearchCV* bekerja dengan cara mencoba semua kombinasi nilai *hyperparameter* pada model hingga didapatkan nilai *hyperparameter* yang memberikan hasil kinerja model terbaik. Nilai *hyperparameter* yang dilakukan pencarian dengan *Grid SearchCV* adalah nilai k.

Berikut adalah perhitungan batas maksimal nilai k menggunakan persamaan 4 :

$$K = \sqrt{N}$$
$$= \sqrt{360}$$
$$= 18{,}97 \approx 19$$

Berdasarkan hasil perhitungan persamaan 4, rentang nilai k sebagai *hyperparameter* adalah 1 hingga 19. Selanjutnya dilakukan *hyperparameter tunning* menggunakan *Grid SearchCV* terhadap 360 *data testing* dengan nilai *hyperparameter cross validation* yang akan dibandingkan adalah 3, 5, dan 10.

TABEL VI
NILAI HYPERPARAMETER TERBAIK SETIAP METODE

Metode Klasifikasi Multi-Label	Cross Validation	Nilai Hyperparameter k terbaik
Binary Relevance - KNN	3	2
	5	3
	10	2
Label Powerset - KNN	3	17
	5	18
	10	17
Classifier Chain - KNN	3	9
	5	9
	10	8
Multi Label - KNN	3	4
	5	8
	10	8

Tabel 3 menunjukkan nilai *hyperparameter* k terbaik hasil dari pencarian *hyperparameter* terbaik menggunakan *Grid SearchCV* dengan mencoba *hyperparameter cross validation* 3, 5, dan 10. Nilai *hyperparameter* k terbaik hasil dari *Grid SearchCV* disimpan ke dalam *best parameter*. Nilai *hyperparameter* terbaik digunakan sebagai *hyperparameter* pada model klasifikasi multi-label setiap metode. Setelah nilai *hyperparameter* ditentukan, dilakukan evaluasi hasil pengujian model klasifikasi terhadap 90 data *testing*.

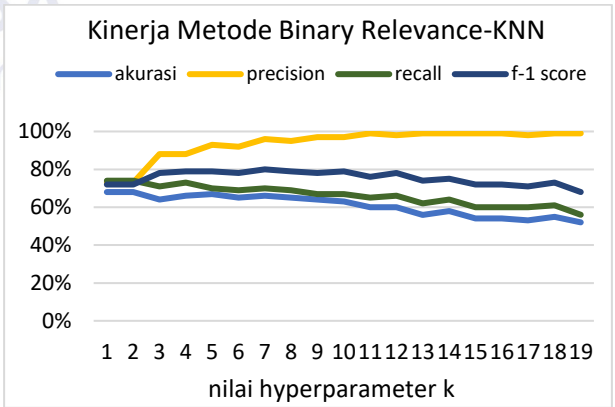
E. *Evaluation*

Evaluasi dilakukan untuk mengetahui kinerja model yang telah dibangun dalam melakukan klasifikasi multi-label pertanyaan Kotakode. Evaluasi kinerja metode dilakukan menggunakan perhitungan manual nilai *precision*, *recall* dan *f-1 score* berdasarkan *confusion matrix* dan menggunakan fungsi *classification report* dari *library scikit-learn python*. Selain itu nilai akurasi didapatkan dengan memanggil fungsi *accuracy score library scikit learn*.

Berikut adalah hasil evaluasi dari setiap pengujian metode yang digunakan pada klasifikasi multi-label :

1. *Problem Transformation Binary Relevance* – KNN

   Nilai akurasi metode *Problem Transformation Binary Relevance*-KNN terbaik didapatkan ketika menggunakan nilai k=2 dan cv=10.



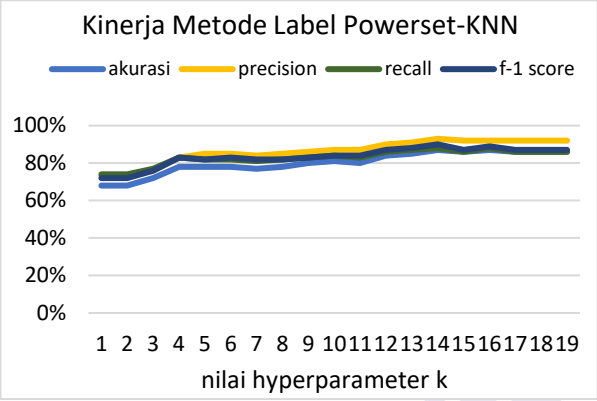
Gbr. 7 Grafik pengujian metode *Binary Relevance*-KNN

Gbr 7 menunjukkan hasil pengujian metode *Binary Relevance*-KNN terhadap data *testing.* Pengujian tersebut dilakukan pada setiap nilai k rentang nilai 1 sampai 19 dengan 10 *cross validation*. Hasil pengujian metode

Problem Transformation Binary Relevance-KNN terhadap data *testing* menghasilkan nilai akurasi 0.68. Artinya 68% data berhasil diklasifikasikan secara tepat oleh model yang dibangun menggunakan metode *Problem Transformation Binary Relevance* dengan nilai *hyperparameter* k=2 pada algoritma KNN. Sedangkan nilai *precision, recall,* dan *f1-score* adalah 0.73, 0.74, dan 0.72. Nilai *hyperparameter* k=2 dipilih berdasarkan hasil pencarian nilai *hyperparameter* terbaik menggunakan *Grid SearchCV.*

2. *Problem Transformation Label Powerset* – KNN

Nilai akurasi metode *Problem Transformation Label Powerset*-KNN terbaik didapatkan ketika menggunakan nilai k=17 dan cv=10.



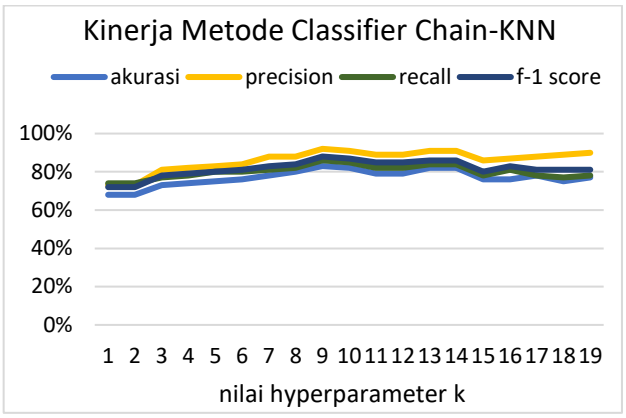
Gbr. 8 Grafik pengujian metode *Label Powerset*-KNN

Gbr 8 menunjukkan hasil pengujian metode *Label Powerset* -KNN terhadap data *testing.* Pengujian tersebut dilakukan pada setiap nilai k rentang nilai 1 sampai 19 dengan 10 *cross validation*. Hasil pengujian metode *Problem Transformation Label Powerset*-KNN terhadap data *testing* menghasilkan nilai akurasi 0.86. Artinya 86% data berhasil diklasifikasikan secara tepat oleh model yang dibangun menggunakan metode *Problem Transformation Label Powerset* dengan nilai *hyperparameter* k=17 pada algoritma KNN. Sedangkan nilai *precision, recall,* dan *f1-score* adalah 0.92, 0.86, dan 0.87. Nilai *hyperparameter* k=17 dipilih berdasarkan hasil pencarian nilai *hyperparameter* terbaik menggunakan *Grid SearchCV.*

3. *Problem Transformation Classifier Chain* – KNN

Nilai akurasi metode *Problem Transformation Classifier Chain* – KNN terbaik didapatkan ketika menggunakan nilai nilai k=9 dan cv=3.

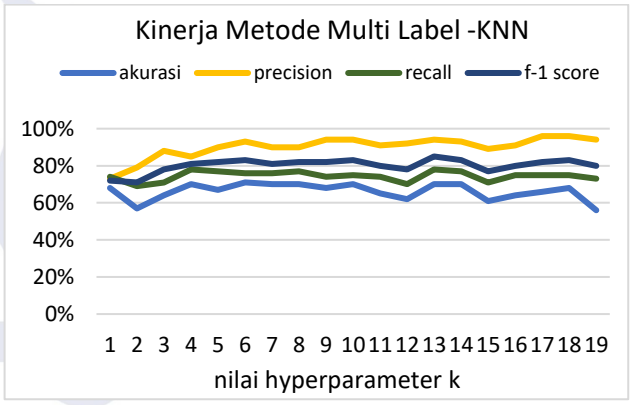
Gbr 9 menunjukkan hasil pengujian metode *Classifier Chain*-KNN terhadap data *testing.* Pengujian tersebut dilakukan pada setiap nilai k rentang nilai 1 sampai 19 dengan 3 *cross validation*. Hasil pengujian metode *Problem Transformation Classifier Chain* – KNN terhadap data testing menghasilkan nilai akurasi 0.83. Artinya 83% data berhasil diklasifikasikan secara tepat oleh model yang dibangun menggunakan metode *Problem Transformation Classifier Chain* dengan nilai *hyperparameter* k=9 pada algoritma KNN. Sedangkan nilai *precision, recall,* dan *f1-score* adalah 0.92, 0.86, dan 0.87. Nilai *hyperparameter* k=9 dipilih berdasarkan hasil pencarian nilai *hyperparameter* terbaik menggunakan *Grid SearchCV.*



Gbr. 9 Grafik pengujian metode *Classifier Chain*-KNN

4. *Algorithm Adaptation* ML-KNN

Nilai akurasi metode *Algorithm Adaptation* ML-KNN terbaik didapatkan ketika menggunakan nilai *hyperparameter* k=8 dan cv=5.



Gbr. 10 Grafik pengujian metode Multi Label-KNN

Gbr 10 menunjukkan hasil pengujian metode *Algorithm Adaptation*-KNN terhadap data *testing.* Pengujian tersebut dilakukan pada setiap nilai k rentang nilai 1 sampai 19 dengan 5 *cross validation*. Hasil pengujian metode *Algorithm Adaptation* ML-KNN terhadap data *testing* menghasilkan nilai akurasi 0.70. Artinya 70% data berhasil diklasifikasikan secara tepat oleh model yang dibangun menggunakan metode *Algorithm Adaptation* ML-KNN dengan nilai *hyperparameter* k=8. Sedangkan nilai *precision, recall,* dan *f1-score* adalah 0.90, 0.77, dan 0.82. Nilai *hyperparameter* k=8 dipilih berdasarkan hasil dari penentuan nilai *hyperparameter* terbaik menggunakan *Grid SearchCV.*

F. Perbandingan Metode Problem Transformation-KNN dengan *Algorithm Adaptation* ML-KNN

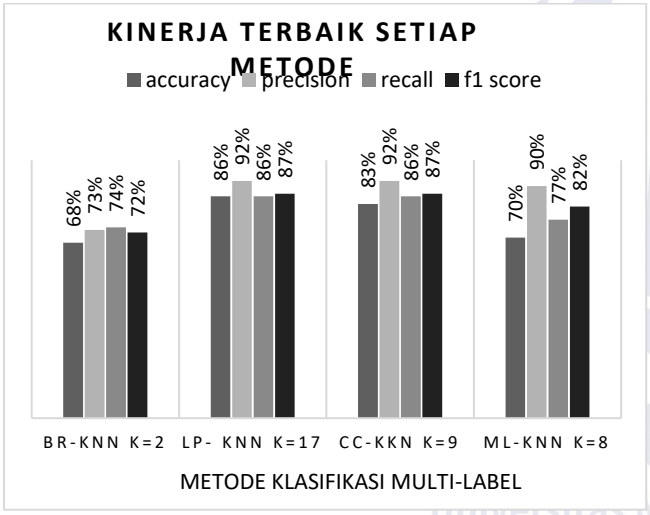
Pada penelitian ini dilakukan perbandingan antara metode *Problem Transformation*-KNN dengan *Algorithm Adaptation*

ML-KNN. Nilai kinerja metode yang dibandingkan adalah nilai akurasi, *precision*, *recall*, dan *f-1 score*. Tabel VII menunjukkan perbandingan kinerja antara metode *Problem Transformation Binary Relevance, Label Powerset, Classifier Chain*, dengan algoritma klasifikasi *K-Nearest Neighbor* dan metode *Algorithm Adaptation* ML-KNN.

TABEL VII
HASIL PENGUJIAN SETIAP METODE

Metode	Akurasi	Precision	Recall	F1-score
Binary Relevance - KNN	68%	73%	74%	72%
Label Powerset - KNN	86%	92%	86%	87%
Classifier Chain - KNN	83%	92%	86%	87%
Multi Label - KNN	70%	90%	77%	82%

Berdasarkan Tabel VII, metode pendekatan *Problem Transformation Label Powerset* dengan algoritma klasifikasi *K-Nearest Neighbor* menghasilkan nilai akurasi, *precision, recall*, dan *f1-score* terbaik yaitu 86%, 92%, 86%, dan 87%. Sehingga, metode *Problem Transformation Label Powerset*-KNN menghasilkan kinerja lebih baik dibandingkan dengan metode *Algorithm Adaptation* ML-KNN dalam melakukan klasifikasi multi-label pertanyaan Kotakode.



Gbr. 11 Grafik perbandingan kinerja metode

IV. KESIMPULAN

Berdasarkan pembahasan dari hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa klasifikasi multi-label pertanyaan Kotakode dengan tagar sebagai label atau kelas dapat dilakukan menggunakan metode *Problem Transformation* dengan algoritma klasifikasi *K-Nearest Neighbo*r (KNN) dan *Algoritm Adaptation* Multi-Label KNN (ML-KNN). Hasil kinerja klasifikasi multi-label terbaik didapatkan ketika mengunakan metode *Problem Transformation Label Powerset* dengan algoritma klasifikasi KNN. Metode *Problem Transformation Label Powerset*-KNN menghasilkan nilai akurasi, *precision, recall*, dan *f1 score* terbaik yaitu 86%, 92%, 86%, dan 87%. Sehingga, metode *Problem Transformation Label Powerset*-KNN menghasilkan nilai kinerja lebih baik apabila dibandingkan dengan *Algoritm Adaptation* ML-KNN dalam melakukan klasifikasi multi-label pertanyaan Kotakode.

Saran yang dapat diberikan untuk penelitian topik klasifikasi multi-label selanjutnya adalah menggunakan metode klasifikasi lain sebagai bahan perbandingan kinerja dengan jumlah data lebih banyak. Selain itu, apabila dataset *imbalance* diharapkan teknik *oversampling*, *undersampling* ataupun teknik lain dalam menangani permasalahan *imbalance* dataset dapat diimplementasikan.

REFERENSI

[1] B. G. Putra dan N. Rochmawati, "Klasifikasi Berdasarkan Question Dalam Stack Oerflow Menggunakan Algoritma Naïve Bayes," vol. 02, hal. 259–267, 2021.

[2] T. P. Sahu, R. S. Thummalapudi, dan N. K. Nagwani, "Automatic Question Tagging Using Multi-label Classification in Community Question Answering Sites," *Proc. - 6th IEEE Int. Conf. Cyber Secur. Cloud Comput. CSCloud 2019 5th IEEE Int. Conf. Edge Comput. Scalable Cloud, EdgeCom 2019*, hal. 63–68, 2019, doi: 10.1109/CSCloud/EdgeCom.2019.00-17.

[3] Z. Abdallah, A. El-Zaart, dan M. Oueidat, "Comparison of multilabel problem transformation methods for text mining," in *2015 Fifth International Conference on Digital Information and Communication Technology and its Applications (DICTAP)*, Apr 2015, hal. 115–118, doi: 10.1109/DICTAP.2015.7113182.

[4] K. A. Nugraha dan H. Herlina, "Klasifikasi Pertanyaan Bidang Akademik Berdasarkan 5W1H menggunakan K-Nearest Neighbors," *J. Edukasi dan Penelit. Inform.*, vol. 7, no. 1, hal. 44, 2021, doi: 10.26418/jp.v7i1.45322.

[5] M. O. Ibrohim dan I. Budi, "Multi-label Hate Speech and Abusive Language Detection in Indonesian Twitter," in *Proceedings of the Third Workshop on Abusive Language Online*, 2019, hal. 46–57, doi: 10.18653/v1/W19-3506.

[6] A. Wiraguna, S. Al Faraby, dan Adiwijaya, "Klasifikasi Topik Multi Label pada Hadis Bukhari dalam Terjemahan Bahasa Indonesia Menggunakan Random Forest," *e-Proceeding Eng.*, vol. 6, no. 1, hal. 2144–2153, 2019.

[7] S. C. Dharmadhikari, M. Ingle, dan P. Kulkarni, "A Comparative Analysis of Supervised Multi-label Text Classification Methods," *Int. J. Eng. Res. Appl.*, vol. 1, no. 4, hal. 1952–1961, 2011.

[8] A. Hanafi, A. Adiwijaya, dan W. Astuti, "Klasifikasi Multi Label pada Hadis Bukhari Terjemahan Bahasa Indonesia Menggunakan Mutual Information dan k-Nearest Neighbor," *J. Sisfokom (Sistem Inf. dan Komputer)*, vol. 9, no. 3, hal. 357–364, 2020, doi: 10.32736/sisfokom.v9i3.980.

[9] O. Maimon dan L. Rokach, *Data Mining and Knowledge Discovery Handbook*. Boston, MA: Springer, Boston, MA, 2010.

[10] O. W. Purbo, *Text Mining Analisis MedSos, Kekuatan Brand & Intelegen di Internet*. Penerbit Andi, 2019.

[11] A. B. Hassanat, M. A. Abbadi, G. A. Altarawneh, dan A. A. Alhasanat, "Solving the Problem of the K Parameter in the KNN Classifier Using an Ensemble Learning Approach," vol. 12, no. 8, hal. 33–39, 2014, [Daring]. Tersedia pada: http://arxiv.org/abs/1409.0919.

[12] L. Hamel, *Knowledge Discovery with Support Vector Machines*. 2009.

[13] F. Herrera, F. Charte, A. J. Rivera, dan M. J. del Jesus, *Multilabel Classification*. Cham: Springer International Publishing, 2016.

[14] J. Han, M. Kamber, dan J. Pei, *Data Mining: Concepts and Techniques Thrid Edition*. Morgan Kaufman Publishers, 2012.

[15] L. Perkovic, *Introduction to Computing Using Python*. Wiley Publishing, 2012.

[16] M. A. Hamada dan L. Naizabayeva, "Decision Support System with K-Means Clustering Algorithm for Detecting the Optimal Store Location Based on Social Network Events," *2020 IEEE Eur. Technol. Eng. Manag. Summit, E-TEMS 2020*, 2020, doi: 10.1109/E-TEMS46250.2020.9111758.