# Kategorisasi Berita *Multi-Label* Berbahasa Indonesia Menggunakan Algoritma *Random Forest*

**Brama Hendra Mahendra[1], Adiwijaya[2], Untari Novia Wisesty[3]**

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[1]bramahendramahendra@students.telkomuniversity.ac.id, [2]adiwijaya@telkomuniversity.ac.id,
[3]untarinw@staff.telkomuniversity.ac.id

**Abstrak**

**Berita merupakan informasi mengenai sesuatu yang sedang terjadi atau sudah terjadi. Seiring dengan berkembangnya teknologi dimana berita disajikan dalam bentuk *website* karena hal itu menyebabkan jumlah berita digital yang dirilis oleh beberapa portal berita setiap harinya menjadi sangat banyak. Dari banyaknya ketersediaan dokumen berita yang ada, berdampak pada banyaknya dokumen berita yang memiliki makna yang sama. Berdasarkan dari uraian diatas dibutuhkan metode-metode pengkategorian berita yang baik untuk memudahkan dalam pengambilan informasi. Dalam hal ini, banyak metode yang dapat dilakukan dalam mengkategorikan berita salah satunya dengan metode *Random Forest.* Tapi sebelum menggunakan metode tersebut, terdapat beberapa langkah yang harus dilakukan dalam menentukan nilai dari *random forest.* Salah satu langkah yang harus dilakukan adalah menentukan *feature extraction* dengan metode *Regular Expression* dan dilanjutkan dengan pembobotan *TF-IDF* lalu setelah itu melakukan *Cross-Validation* dengan *k-Fold*. Dataset yang digunakan terdiri dari dua jenis yaitu data *testing* dan data *training*. Untuk hasil uji coba diperoleh nilai sebesar 0,126 dari proses persamaan *Hamming Loss*.**

**Kata kunci: Random Forest, Dokumen Berita, Feature Extraction, Pembobotan TF-IDF, Cross-Validation, Hamming Loss**

**Abstract**

**News is an information about something that is happening or has happened. Along with the development of technology where news is presented in the form of websites, it causes a large number of digital news released by several news portals issued. From the abundance of news document that exist, it has an impact on the number of news document that have the same meaning. Based on the description above, it needs a good news categorization methods to facilitate information retrieval. In this case, there are many methods can be done in categorizing the news one of them by the Random Forest method. But before using this method, there are several steps must be taken to determine the value of random forest. One step that must be done is to determine feature extraction using the Regular Expression method and complete it by weighting of TF-IDF, and then doing Cross-Validation. The data used consists of two types, testing data and training data. For the results of the trial obtained a value of 0.126 from the Hamming Loss equation process.**

***Keywords: Random Forest, News Document, Feature Extraction, Weighting of TF-IDF, Cross-Validation, Hamming Loss***

## 1. Pendahuluan

Klasifikasi teks adalah proses mengklasifikasikan teks secara sistematis. Contoh dalam penerapan klasifikasi teks seperti penyaringan teks, klasifikasi berita, mencari informasi menarik di web, dan lain-lain [1]. Berita merupakan suatu informasi yang akan selalu diperbarui. Dimana isi kandungan dari berita berupa informasi baru yang belum terkandung dalam berita sebelumnya. Dalam hal ini, berita sering disebut aliran informasi yang dinamis. Selain itu, berita banyak disajikan pada bentuk teks pada media digital dan biasanya dikelompokkan berdasarkan isi dari berita tersebut. Isi dari beritapun berupa teks, dimana berita disimpan dan dikelompokkan sesuai dengan kelompoknya.

Permasalahan yang sering muncul adalah penggunaan media digital yang banyak dalam penyampaian informasi menyebabkan jumlah berita digital yang dirilis oleh beberapa portal berita setiap harinya menjadi sangat banyak, dan menyebabkan berita memiliki keterkaitan lebih dari satu kategori. Dari banyaknya ketersediaan berita yang ada dan memiliki keterkaitan dengan banyak kategori atau *multi-label* berdampak pada banyaknya berita yang memiliki makna yang sama dan selalu terkait dengan kata yang ambigu, dimana satu obyek berita memiliki sejumlah kelas kategori yang berbeda.

Berdasarkan dari uraian diatas dibutuhkan metode pengklasifikasian dan pengorganisasian dokumen berdasarkan isinya, yaitu *text mining* dan salah satu algoritma *machine learning* yang dapat digunakan dalam *text categorization* adalah *Random Forest* [2] [3]

Penggunaan algoritma *Random Forest* dalam proses pengklasifikasian data dalam jumlah besar yaitu melakukan penggabungan pohon dengan melakukan *training* pada sampel data yang dimiliki. Dalam hal ini, semakin banyak pohon akan mempengaruhi akurasi yang akan didapat menjadi lebih baik.

Berdasarkan masalah diatas, permasalahan yang akan dibahas adalah bagaimana melakukan klasifikasi teks dan menentukan pembobotan dalam suatu dokumen berita, dan bagaimana menganalisa tingkat akurasi performansi yang dihasilkan dari proses klasifikasi.

Tujuan yang ingin dicapai dalam penelitian ini antara lain dapat melakukan klasifikasi teks berita dengan menggunakan metode *random forest*, dapat menentukan pembobotan untuk dokumen berita dengan menggunakan pembobotan *TF-IDF*, dan dapat menganalisa tingkat akurasi performansi dengan menggunakan persamaan *Hamming Loss* yang telah dihasilkan dari proses klasifikasi.

## 2. Studi Terkait

Klasifikasi teks untuk pengorganisasian berita dalam memudahkan pembaca dalam pengambilan informasi [2]. Dari banyak berita yang tersedia di situs web pengorganisasian berita merupakan suatu hal yang penting, karena dapat memudahkan dalam membaca suatu berita yang diinginkan tanpa harus melihat semua berita.

Berita memiliki aliran yang dinamis dimana informasi yang terkandung didalamnya adalah informasi baru dan belum ada di berita lainnnya. Selain itu, berita dapat memiliki lebih dari satu kategori atau bisa disebut *multi-label*. Dalam hal ini, seringkali kesulitan untuk mengkategorikan suatu berita dikarenakan data yang dilihat terlihat ambigu dikarenakan satu objek berita memiliki kategori yang berbeda.

Penelitian seperti ini pernah dilakukan sebelumnya terkait klasifikasi *multi-label* terhadap data teks, baik data tersebut berita atau bukan dan menggunakan Bahasa Indonesia atau bukan yaitu *Naïve Bayes Classifier* [4]. Dimana dalam penelitian tersebut memiliki tingkat akurasi 68,64% dari jumlah 524 dokumen berita yang di uji coba. Selain itu, Sigit Bagus Setiawan juga telah melakukan Klasifikasi Topik Berita Berbahasa Indonesia menggunakan *Weighted K-Nearest Neighbor* [5]. Dimana penelitian ini ditujukan agar dapat membuat sistem yang mampu mengkategorikan setiap berita berbahasa Indonesia pada kelas yang seharusnya. Pemilihan pemberian bobot pada *weighted k-nearest neighbor* cukup berpengaruh atas performa sistem. Dan semakin besar nilai *k* pada awalnya akan memperbesar persentase kebenaran pada klasifikasi, kemudian jika *k* sudah pada titik optimum maka besar persentase akan cenderung turun. Pada penelitian tersebut diperoleh akurasi sebesar 75,86%. Pada penelitian tersebut juga menyimpulkan bahwa hasil dari data *training* sangat berpengaruh untuk pembelajaran sistem, sehingga akurasi terbaik pada penelitian tersebut dimana data *training* dan data *testing* adalah 80% berbanding 20%. Selain daripada jumlah data *training*, parameter *k* dan pembobotan pada metode klasifikasi *weighted k-NN* mempengaruhi akurasi sistem.

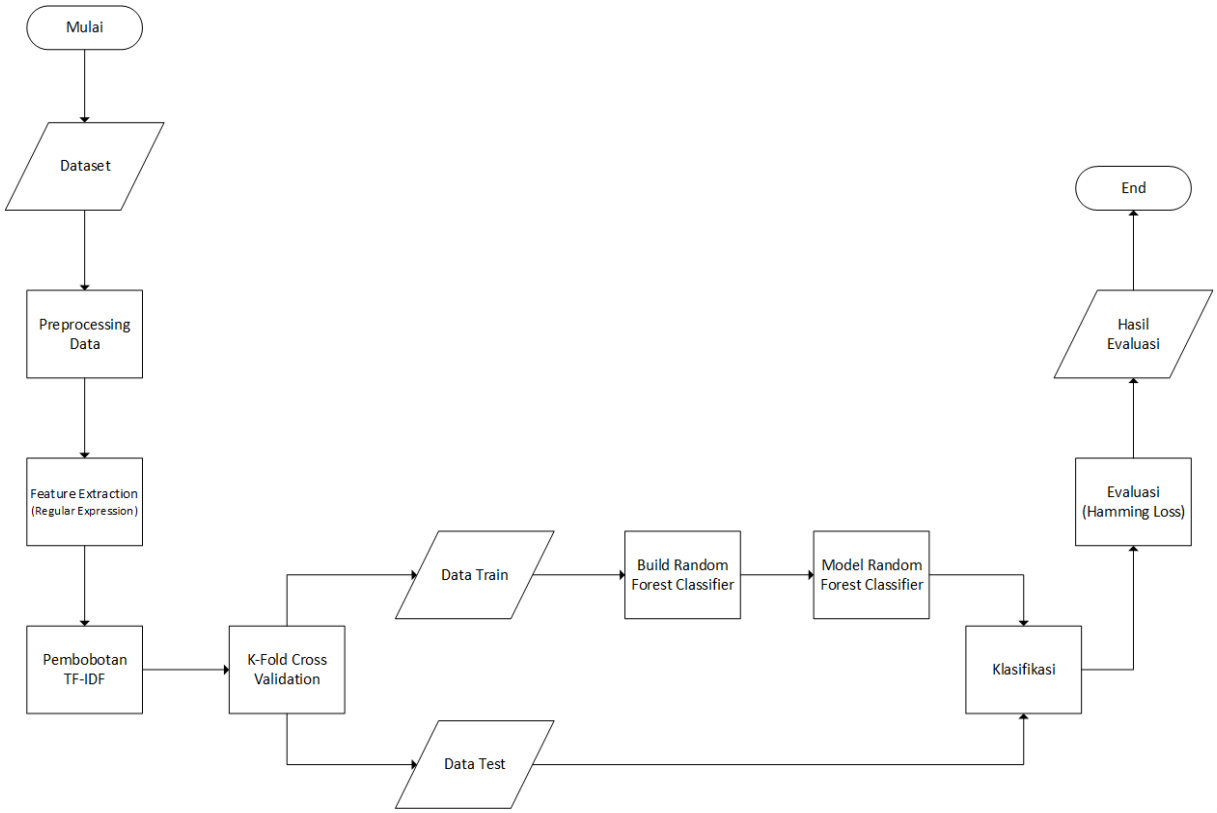
Selain itu beberapa kasus lagi mengenai klasifikasi, seperti pada penelitian yang dilakukan oleh Komang Ari Widani, metode yang digunakan adalah metode *Chi-Square* dan *multinomial naïve bayes* yang memperoleh akurasi sebesar 85,28%. Pada penelitian tersebut penggunaan tanpa *stemming* menghasilkan akurasi yang lebih baik dibandingkan menggunakan *stemming* dan menggunakan seleksi fitur yaitu *Chi-Square* belum mampu untuk meningkatkan performansi *Multinomial Naive Bayes Classifier*, karena semakin kecil level signifikansi yang digunakan pada seleksi fitur membuat semakin sedikit pula jumlah fitur yang digunakan untuk *learning model* klasifikasi. Tetapi penggunaan seleksi fitur *Chi-Square* mampu mengurangi waktu komputasi model klasifikasi *Multinomial Naive Bayes* dengan kecepatan yang dapat ditingkatkan yaitu sebesar 6%-34% [6]. Pada penelitian yang dilakukan oleh Fahmi Salman Nurfikri, metode yang digunakan pada topik berita yaitu menggunakan metode *mutual information* dan *bayesian network* yang menghasilkan akurasi sebesar 75,34% dimana fitur seleksi *Mutual Information* mampu meningkatkan akurasi dari klasifikasi *Bayesian Network* [7]. Siti Nur Asiyah dan Kartika Fithriasari melakukan penelitian tentang Klasifikasi Berita Online Menggunakan Metode *Support Vector Machine* dan *K-Nearest Neighbor* [8]. Kedua metode ini akan dibandingkan untuk mengetahui hasil ketepatan klasifikasi yang paling baik. Hasil dari penelitian ini bahwa *SVM kernel linier* dan *kernel polynomial* menghasilkan ketepatan klasifikasi yang paling baik adalah *kernel polynomial*.

Berdasarkan permasalahan yang disebutkan, diusulkan sebuah metode baru untuk mengkategorikan dokumen berita yang isinya terkait dengan banyak kategori, metode ini berbasis perhitungan kata spesifik yaitu perhitungan kata-perkata dari isi dokumen terhadap jenis kategorinya. Metode ini nantinya akan diuji untuk mengkategorikan dokumen berita Berbahasa Indonesia dengan kategori yang berbeda-beda agar mempermudah pembaca dalam mencari berita.

## 3. Sistem yang Dibangun

3.1 Gambaran Umum Perancangan Sistem

Pada tahap ini akan dijelaskan tentang sistem yang akan dibangun. Berikut adalah *flowchart* yang menggambarkan alur kerja pembangunan sistem klasifikasi teks berita secara umum.




**Gambar 1. Gambaran Umum Sistem.**

3.2 Dataset

Data yang digunakan merupakan dokumen berita yang didapatkan dari situs web berita nasional. Adapun, salah satu situs berita yang digunakan yaitu https://www.jawapos.com. Pada situs tersebut dikumpulkan 500 data berita, dimana data tersebut di klasifikasi ke dalam beberapa sub kategori berita. Data tersebut akan diberikan label pada saat pengkategorisasian berita. Berikut merupakan contoh dataset yang akan digunkan adalah sebagai berikut.

**Tabel 1. Contoh Dataset Berita**

| Kelas | Politik | Ekonomi | Teknologi |
|---|---|---|---|
| Ketua Tim Kampanye Nasional (TKN) Jokowi-Ma'ruf, Erick Thohir optimistis, capres petahana akan moncer hadapi Prabowo di debat keempat. | 1 | 0 | 0 |
| Perdagangan pasar saham Indonesia hari ini berpeluang untuk berbalik arah menguat. | 0 | 1 | 0 |
| Perangkat storage Dell EMC Data Domain dan Integrated Data Protection Appliance (IDPA) mendapat pembaruan. | 0 | 0 | 1 |

Pada tabel 1 dijelaskan bahwa jika berisi nilai 1 pada kolom tersebut maka termasuk dalam kelas tersebut, sedangkan sebaliknya, jika berisi nilai 0 pada kolom tersebut berarti tidak termasuk kelas tersebut. Pelabelan tersebut diperoleh dari kelas berita yang terdapat pada website jawapos.

3.3 *Pre-Processing*

Struktur data yang baik dapat memudahkan proses komputerisasi secara otomatis. Oleh karena itu, diperlukan proses pengubah bentuk menjadi data yang terstruktur sesuai kebutuhannya untuk proses *data mining*, yang biasanya akan menjadi nilai-nilai numerik. Proses ini sering disebut *Text Processing* atau *Pre-Processing*. Setelah data terstruktur dan berupa nilai numerik maka data dapat dijadikan sebagai sumber data yang dapat diolah lebih lanjut [9]. Pada proses *Pre-Processing* dilakukan 4 tahapan yaitu sebagai berikut.

3.3.1 *Punctuation Removal*

Pada tahap ini, dilakukan proses *Punctuation Removal*. *Punctuationl removal* adalah proses penghilang tanda baca yang ada pada teks. Adapun implementasi dari proses *punctuation removal* dapat dilihat pada tabel dibawah ini.

**Tabel 2. Contoh proses *Punctuation Removal***

| | |
|---|---|
| ***Input*** | Garuda Indonesia membeberkan alasan tarif tiket pesawat menjadi mahal belakangan ini. Penyebab utamanya adalah perang harga yang sudah dilakukan selama tiga hingga lima tahun terakhir. Para maskapai memberikan harga di bawah cost mereka. Akibatnya, kinerja keuangan pun berdarah-darah alias rugi. |
| ***Output*** | Garuda Indonesia membeberkan alasan tarif tiket pesawat menjadi mahal belakangan ini Penyebab utamanya adalah perang harga yang sudah dilakukan selama tiga hingga lima tahun terakhir Para maskapai memberikan harga di bawah cost mereka Akibatnya kinerja keuangan pun berdarah darah alias rugi |

3.3.2 *Case Folding*

Pada tahap selanjutnya, dilakukan proses *Case Folding*. *Case folding* adalah mengubah semua huruf dalam dokumen menjadi huruf kecil. Adapun implementasi dari proses *case folding* dapat dilihat pada tabel dibawah ini.

**Tabel 3. Contoh proses *Case Folding***

| | |
|---|---|
| ***Input*** | Garuda Indonesia membeberkan alasan tarif tiket pesawat menjadi mahal belakangan ini Penyebab utamanya adalah perang harga yang sudah dilakukan selama tiga hingga lima tahun terakhir Para maskapai memberikan harga di bawah cost mereka Akibatnya kinerja keuangan pun berdarah darah alias rugi |
| ***Output*** | garuda indonesia membeberkan alasan tarif tiket pesawat menjadi mahal belakangan ini penyebab utamanya adalah perang harga yang sudah dilakukan selama tiga hingga lima tahun terakhir para maskapai memberikan harga di bawah cost mereka akibatnya kinerja keuangan pun berdarah darah alias rugi |

3.3.3 *Stopword Removal*

Pada tahap ini, dilakukan proses *Stopword Removal*. *Stopword removal* adalah kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan *bag-of-words* [10]. Kata-kata tersebut dihilangkan dan tidak akan diproses pada tahap selanjutnya karena dianggap memiliki makna yang kurang penting dan pengaruh yang tidak begitu besar dalam proses klasifikasi yang dilakukan. Adapun implementasi dari proses *Stopword Removal* dapat dilihat pada tabel dibawah ini.

**Tabel 4. Contoh proses *Stopword Removal***

| | |
|---|---|
| ***Input*** | garuda indonesia membeberkan alasan tarif tiket pesawat menjadi mahal belakangan ini penyebab utamanya adalah perang harga yang sudah dilakukan selama tiga hingga lima tahun terakhir para maskapai memberikan harga di bawah cost mereka akibatnya kinerja keuangan pun berdarah darah alias rugi |
| ***Output*** | garuda indonesia membeberkan alasan tarif tiket pesawat mahal penyebab utamanya perang harga tahun maskapai harga cost akibatnya kinerja keuangan berdarah darah alias rugi |

3.3.4 *Stemming Data*

Pada tahap selanjutnya, dilakukan proses *Stemming Data*. *Stemming data* adalah tahap mencari *root* kata dari tiap kata. Pada tahap ini dilakukan proses pengambilan berbagai bentukan kata kedalam suatu representasi yang sama. Adapun implementasi dari proses *stemming data* dapat dilihat pada tabel dibawah ini.

**Tabel 5. Contoh proses *Stemming Data***

| | |
|---|---|
| ***Input*** | garuda indonesia membeberkan alasan tarif tiket pesawat mahal penyebab utamanya perang harga tahun maskapai harga cost akibatnya kinerja keuangan berdarah darah alias rugi |
| ***Output*** | garuda indonesia beber alas tarif tiket pesawat mahal sebab utama perang harga tahun maskapai harga cost akibat kerja uang darah darah alias rugi |

3.4 *Feature Extraction (Regular Expression)*

Dalam tahap ini, akan dilakukan *Feature Extraction*. *Feature extraction* adalah proses menghasilkan fitur yang akan digunakan pada proses klasifikasi [11]. Dalam hal ini, *feature extraction* menggunakan metode *Regular Expression*. Metode *regular expression* merupakan sekumpulan kata yang diberikan dalam sebuah paragraf dan ketika menghitung *regular expression* biasanya dilakukan dekonstruksi untuk mencocokkan teks berdasarkan pola tertentu, terutama untuk kasus-kasus kompleks [10].

Untuk melakukan proses *Regular Expression* ditentukan terlebih dahulu kata kunci untuk masing-masing kelas. Kata kunci ini diperoleh dari jumlah kata yang sering muncul pada dokumen berita. Untuk menentukan kata kunci tersebut menggunakan bantuan dari website http://id.wordcounter360.com, dimana website tersebut menginputkan dokumen berita dan mengoutputkan jumlah setiap kata yang sering muncul pada dokumen berita tersebut. Hasil output tersebut menjadi kata kunci pada masing-masing kelas. Setelah dilakukan proses menentukan kata kunci dari masing-masing kelas, maka dilakukan proses *regular expression.* Adapun implementasi *regular expression* memiliki beberapa tahapan yaitu.

3.4.1 *Tagger*

**Tabel 6. Contoh proses *Tagger***

| | |
|---|---|
| ***Input*** | menteri uang sri mulyani indrawati tarik atur menteri uang pmk nomor pmk laku paja transaksi dagang |
| ***Output*** | [[('menteri', 'NN'), ('uang', 'NN'), ('sri', 'NN'), ('mulyani', 'VB'), ('indrawati', 'VB'), ('tarik', 'NN'), ('atur', 'NN'), ('menteri', 'NN'), ('uang', 'NN'), ('pmk', 'NN'), ('nomor', 'NN'), ('pmk', 'NN'), ('laku', 'NN'), ('paja', 'NN'), ('transaksi', 'NN'), ('dagang', 'NN')]] |

3.4.2 Fitur *One Doc*

**Tabel 7. Contoh proses fitur *one doc***

| | |
|---|---|
| ***Input*** | [[('menteri', 'NN'), ('uang', 'NN'), ('sri', 'NN'), ('mulyani', 'VB'), ('indrawati', 'VB'), ('tarik', 'NN'), ('atur', 'NN'), ('menteri', 'NN'), ('uang', 'NN'), ('pmk', 'NN'), ('nomor', 'NN'), ('pmk', 'NN'), ('laku', 'NN'), ('paja', 'NN'), ('transaksi', 'NN'), ('dagang', 'NN')]] |
| ***Output*** | {'menteri': 2, 'uang': 2, transaksi': 1, 'dagang': 1} |

3.4.3 Fitur *All Doc*

**Tabel 8. Contoh proses fitur *all doc***

| | |
|---|---|
| ***Input*** | {'menteri': 2, 'uang': 2, transaksi': 1, 'dagang': 1}<br>{'menteri': 3, 'saham': 1} |
| ***Output*** | {'menteri': 5, 'uang': 2, transaksi': 1, 'dagang': 1, 'saham': 1} |

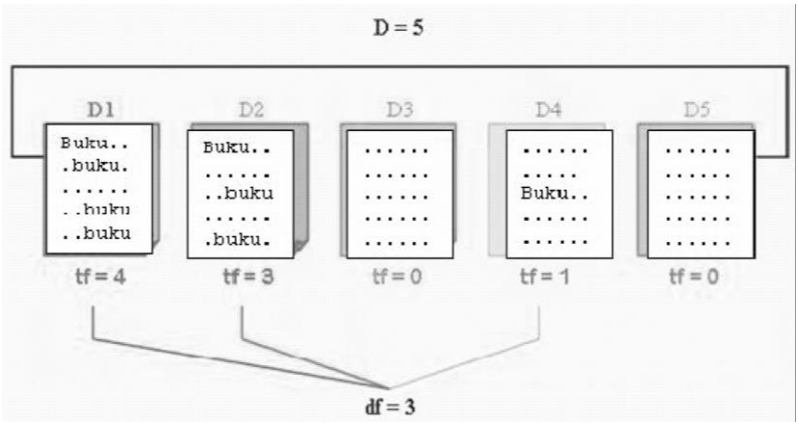
3.5 Pembobotan TF-IDF

Dalam tahap ini, akan dilakukan pembobotan *TF-IDF*. Pembobotan *TF-IDF* adalah salah satu kegiatan yang umumnya bisa dilakukan bertujuan untuk melakukan pembobotan setelah proses *feature extraction* (*regular expression*) dilakukan agar diperoleh hasil angka yang nantinya akan digunakan dalam *cross-validation* [12]. Pembobotan *TF-IDF* adalah jenis pembobotan yang sering digunakan dalam pengambilan informasi dan *text mining*. Pembobotan ini adalah suatu pengukuran statistik untuk mengukur seberapa penting sebuah kata dalam kumpulan dokumen. Tingkat kepentingan meningkat ketika sebuah kata muncul beberapa kali dalam sebuah dokumen tapi diimbangi dengan frekuensi kemunculan kata tersebut dalam sebuah dokumen.

Pada tahap awal proses *TF-IDF* yaitu dihitung terlebih dahulu *Term Frequency* (TF) yaitu frekuensi kemunculan suatu *term* di tiap dokumen. Semakin sering suatu *term* muncul di banyak dokumen maka nilai *IDF*-nya akan kecil. Berikut rumus *TF* [12] yaitu sebagai berikut.

$$TF(t_k, d_j) = f(t_k, d_j) \quad \textbf{(1)}$$

Pada persamaan (1) $f(t_k, d_j)$ adalah frekuensi kemunculan *term* dari dokumen. Dimana $t_k$ adalah *term* dan $d_j$ adalah dokumen. Adapun ilustrasi penggunaan *TF-IDF* yaitu sebagai berikut.




**Gambar 2. Ilustrasi algoritma *TF***

Pada gambar 2, terdapat lima D yaitu D1, D2, D3, D4, dan D5 yang merupakan sebuah dokumen. Dari masing-masing dokumen tersebut diperoleh nilai *Term Frequency* (*TF*) yang berbeda-beda dimana *TF* adalah frekuensi kemunculan suatu *term* di tiap dokumen. Contohnya pada D1 memiliki nilai TF = 4 sedangkan pada D2 memiliki nilai *TF* = 3. Nilai *TF* ini diperoleh dari seberapa banyak kata "buku" muncul dalam suatu dokemen. Pada D1 kata "buku" muncul sebanyak 4 kali yang dimana menghasilkan nilai *TF* pada D1 yaitu 4. Setelah itu, dilakukan proses *DF* yaitu menentukan banyak dokumen yang mengandung kata yang dicari. Contohnya pada gambar 2, diperoleh nilai *DF* = 3 dimana hasil tersebut diperoleh dari banyak nilai *TF* > 0. Pada gambar 2, terdapat nilai *TF* > 0 yaitu pada D1, D2 dan D4. Sedangkan pada D3 dan D5 nilai *TF* = 0. Maka dari hal tersebut nilai *DF* = 3.

Setelah dilakukan proses *TF* selanjutnya dihitung *Inverse Document Frequency* (IDF) yaitu nilai bobot suatu *term* dihitung dari seringnya suatu *term* muncul dibeberapa dokumen. Berikut rumus *IDF* [12] yaitu sebagai berikut.

$$IDF(t_k) = \log \frac{N}{df(t)} \quad \textbf{( 2 )}$$

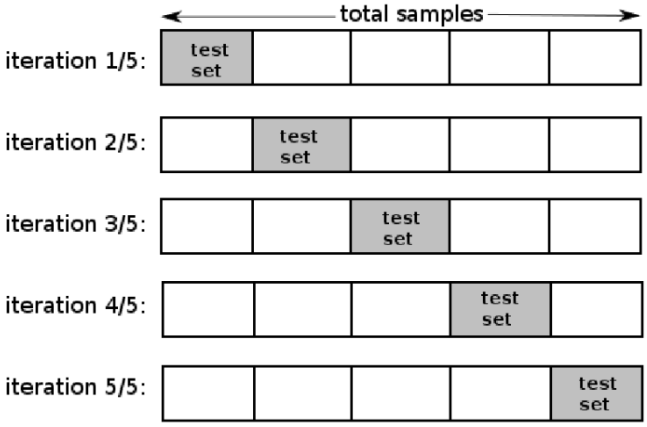
Pada persamaan (2) *N* menyatakan jumlah semua dokumen dan *DF* menyatakan banyak dokumen yang mengandung kata yang dicari untuk lebih jelasnya bisa melihat pada gambar 2. Setelah dilakukan perhitungan *TF* dan *IDF* maka proses selanjutnya yaitu menghitung *TF-IDF*. Pada *TF-IDF* dapat dirumuskan sebagai berikut.

$$TF\ IDF(t_k, d_j) = TF(t_k, d_j) * IDF(d_j) \quad \textbf{( 3 )}$$

Pada persamaan (3) $TF(t_k, d_j)$ menyatakan hasil dari *TF* dan $IDF(d_j)$ menyatakan hasil dari *IDF*. Dimana sebelumnya dihitung terlebih dahulu *Term Frequency* (TF) yaitu frekuensi kemunculan suatu *term* di tiap dokumen. Kemudian dihitung *Inverse Document Frequency* (IDF) yaitu nilai bobot suatu *term* dihitung dari seringnya suatu *term* muncul dibeberapa dokumen.

3.6 *K-Fold Cross Validation*

Setelah dilakukan beberapa proses mulai dari *feature extraction* dengan *regular expression* sampai dengan pembobotan *TF-IDF*, sehingga diperoleh nilai dari setiap fitur tersebut. Selanjutnya dilakukan proses *K-Fold Cross Validation. K-Fold* adalah salah satu metode *Cross-Validation* yang popular dengan melipat data sebanyak *k* dan mengulangi (men-iterasi) eksperimennya sebanyak *k* juga [13]. Dalam *K-Fold Cross-Validation*, data dipartisi ke dalam segmen yang sama, dimana satu segmen dijadikan sebagai data *training*, dan segmen lainnya dijadikan segmen *testing* [14]. Perbedaannya pada metode ini, setiap data yang terdapat di dalam dataset harus pernah dijadikan sebagai data *training* dan juga data *testing*. Dengan *Cross-Validation* seluruh data akan dibagi ke dalam *k* kelompok. Dengan ukuran sejumlah dataset dibagi jumlah *k* yang akan digunakan [15].

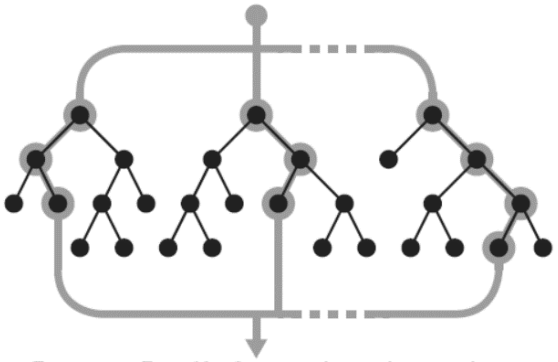



**Gambar 3. 5-*Fold Cross-Validation***

3.6 *Random Forest*

Dalam tahap ini, digunakan metode *Random Forest* yang dilakukan melalui penggabungan pohon dengan melakukan *training* pada sampel data yang dimiliki. Penggunaan metode ini memiliki keuntungan yang kuat dalam hal kompleksitas komputasi, mudah dan cepat dalam penerapannya, khusunya dalam menghasilkan nilai prediksi akurasi, dapat menangani *missing value*, tidak terlalu sensitif pada *outlier* dan dapat menangani sejumlah besar inputan variabel tanpa harus *overfitting*.

Pada *random forest* proses klasifikasi akan berjalan jika semua *tree* telah terbentuk. Pada saaat proses tersebut selesai, inisialisasi dilakukan dengan sebanyak data berdasarkan nilai akurasinya. Keuntungan dalam menggunakan *random forest* adalah mampu mengklasifikasikan data yang memiliki atribut yang tidak lengkap, dan metode ini sangat cocok untuk pengklasifikasian data serta dapat digunakan unutk menangani data sampel yang banyak. Dengan kata lain, dengan menggunakan *random forest* pada klasifikasi akan menghasilkan hasil yang baik. Berikut penjelasan secara umum mengenai *random forest* [16].



**Gambar 4. *Ensambled* pohon-pohon yang digunakan dalam *Random Forest***

3.6 *Hamming Loss*

Setelah dilakukan beberapa proses klasifikasi dengan sistem yang dibangun sebelumnya dari mulai dari *feature extraction* dengan *regular expression*, pembobotan *TF-IDF* dan klasifikasi dengan *Random Forest*, selanjutnya akan dilakukan perhitungan akurasi untuk mengetahui seberapa baik sistem performansi yang dihasilkan. Pada penelitian ini, model perhitungan akurasi yang akan digunakan yaitu menggunakan persamaan *Hamming Loss*. *Hamming Loss* berfungsi untuk menghitung kesalahan pada saat melakukan klasifikasi label yang terdapat pada hasil klasifikasi [17]. Berikut merupakan rumus dari Hamming Loss yaitu sebagai berikut.

$$HL = \frac{1}{NL}\sum_{i=1}^{N}\sum_{j=1}^{L}\left[\hat{Y}_j^{(i)} \neq y_j^{(i)}\right] \quad (4)$$

Pada persamaan (4) *HL* menyatakan nilai persamaan *hamming loss*. Sedangkan, *N* menyatakan jumlah data yang diklasifikasikan, *L* menyatakan jumlah label yang digunakan pada dataset, label prediksi ke-*j* pada data ke-*i* yang dihasilkan dari sistem dinotasikan dengan $\hat{Y}_j^{(i)} dan\ y_j^{(i)}$ adalah label aktual ke-*j* dari data ke-*i* yang didapat dari dataset [18]. Pada penelitian ini hanya menggunakan persamaan *Hamming Loss* karena merupakan matriks yang sering digunakan pada penelitian klasifikasi *multi-label*. Performansi sistem dengan persamaan ini akan

terlihat baik jika nilai yang dihasilkan pada persamaan *Hamming Loss* tersebut mendekati nilai 0, sehingga contoh hasil prediksi klasifikasi pada tabel diatas memberikan makna bahwa sistem performansi yang dibangun cukup baik untuk melakukan klasifikasi terhadap dokumen berita.

## 4. Evaluasi

4.1 Hasil Pengujian

Tujuan dari hasil pengujian sistem adalah untuk mengklasifikasikan berita kedalam 3 label yang ada. Sistem yang dibangun memiliki tiga proses utama yaitu *preprocessing*, *feature extraction* dengan *regular expression*, pembobotan *TF-IDF*, *k-Fold Cross-Validation* dan *random forest classifier*. Hal tersebut dilakukan untuk memperoleh performa yang optimal dari sistem, maka dari itu dilakukan beberapa pengujian.

Pada pengujian pertama ini, saya mencoba untuk membagi menjadi 2 nilai *k* pada *k-Fold* yaitu 4-*Fold* dan 5-*Fold*. Alasan saya membagi menjadi 4-*Fold* dan 5-*Fold* yaitu, agar sesuai dengan *rule of thumb* (aturan umum) dengan proporsi *test set* sebesar 20% dan *train set* sebesar 80% pada saat kondisi 5 –*Fold* dan terkadang juga ada beberapa yang mengujinya saat proporsi *test set* sebesar 25% dan *train set* sebesar 75% pada saat kondisi 4 –*Fold*. Dan selain itu, saya menggunakan dua parameter pada *Random Forest Classifier* yaitu, *n-estimators* adalah jumah pohon pada *random forest* dan *max-depth* adalah parameter untuk menentukan maksimal kedalaman dari suatu pohon. Data yang digunakan pada penelitian ini merupakan data dari *werbsite* berita Jawa Pos dan terdiri dari 500 dokumen berita. Hasil pada percobaan kali ini dapat dilihat pada tabel 9 dan tabel 10 yaitu sebagai berikut.

**Tabel 9. Hasil *Hamming Loss* pada kondisi 4-*Fold***

| n-estimators | Max-depth | | | | | Mean |
|---|---|---|---|---|---|---|
| | 5 | 10 | 15 | 20 | 25 | |
| **10** | 0.14133 | 0.13667 | 0.13333 | 0.13400 | 0.13533 | **0.13613** |
| **20** | 0.13533 | 0.13933 | 0.13400 | 0.13667 | 0.13867 | **0.13680** |
| **30** | 0.14333 | 0.13533 | **0.12933** | **0.12933** | 0.13467 | **0.13440** |
| **40** | 0.14333 | 0.13400 | 0.13400 | 0.13133 | 0.13733 | **0.13600** |
| **50** | 0.14000 | 0.13333 | 0.13467 | 0.13667 | 0.13600 | **0.13613** |
| **60** | 0.14067 | 0.13400 | 0.13600 | 0.13600 | 0.13667 | **0.13667** |
| **70** | 0.14133 | 0.13733 | 0.13533 | 0.13733 | 0.13800 | **0.13787** |
| **80** | 0.14000 | 0.13533 | 0.13467 | 0.13733 | 0.13867 | **0.13720** |
| **90** | 0.14067 | 0.13400 | 0.13267 | 0.13400 | 0.13867 | **0.13600** |
| **100** | 0.14133 | 0.13400 | 0.13400 | 0.13533 | 0.13933 | **0.13680** |
| ***Mean*** | **0.14073** | **0.13533** | **0.13380** | **0.13480** | **0.13733** | |

**Tabel 10. Hasil *Hamming Loss* pada kondisi 5-*Fold***

| n-estimators | Max-depth | | | | | Mean |
|---|---|---|---|---|---|---|
| | 5 | 10 | 15 | 20 | 25 | |
| **10** | 0.14400 | 0.13333 | 0.13133 | 0.13067 | 0.12933 | **0.13373** |
| **20** | 0.13467 | 0.12867 | 0.12800 | 0.12800 | 0.12733 | **0.12933** |
| **30** | 0.13867 | 0.13067 | 0.13000 | **0.12600** | 0.13000 | **0.13107** |
| **40** | 0.13800 | 0.13400 | 0.13067 | **0.12600** | 0.13133 | **0.13200** |
| **50** | 0.14000 | 0.13467 | 0.13400 | 0.13133 | 0.13400 | **0.13480** |
| **60** | 0.13867 | 0.13200 | 0.13000 | 0.12867 | 0.13533 | **0.13293** |
| **70** | 0.13600 | 0.13600 | 0.13333 | 0.12733 | 0.13400 | **0.13333** |
| **80** | 0.13933 | 0.13467 | 0.13267 | 0.13067 | 0.13067 | **0.13360** |
| **90** | 0.14067 | 0.13467 | 0.13200 | 0.13067 | 0.13533 | **0.13467** |
| **100** | 0.13867 | 0.13467 | 0.13267 | 0.13000 | 0.13533 | **0.13427** |
| ***Mean*** | **0.13887** | **0.13333** | **0.13147** | **0.12893** | **0.13227** | |

Pada kedua percobaan yaitu pada tabel 9 dengan kondisi 4-Fold dan pada tabel 10 dengan kondisi 5-Fold dilakukan perhitungan dengan menggunakan 2 parameter pada *random forest* yaitu *n_estimators* = 10, 20, 30, 40,50, 60, 70, 80, 90, 100 dan *max-depth* = 5, 10, 15, 20, 25. Pada percobaan tersebut diperoleh hasil dari persamaan *hamming loss*, contohnya saat kondisi 4-*Ford* lalu nilai *n-estimators* = 10 dan nilai *max-depth* = 5 maka diperoleh hasil persamaan *hamming loss* = 0,14133. Tujuan dari percobaan ini untuk menguji hasil terbaik dari

kondisi 4-*Fold* dan 5-*Fold* dengan mengubah nilai parameter pada *random forest* yaitu *n-estimators* dan *max-depth*.

Pada percobaan tabel 9 diperoleh hasil terbaik dari persamaan *hamming loss* yaitu 0,12933 dimana nilai *k-Fold* = 4, *n-estimators* = 30, dan *max-depth* = 15 dan 20. Sedangkan pada tabel 10 diperoleh hasil terbaik dari persamaan *hamming loss* yaitu 0,12600 dimana nilai *k-Fold* = 5 ,*n-estimators* = 30 dan 40, dan *max-depth* = 20. Jika dibandingkan persamaan *hamming loss* terbaik dari tabel 9 dengan 4-*Fold* dan tabel 10 dengan 5-*Fold* maka nilai persamaan *hamming losss* terbaik yaitu 0,12600 dimana *k-Fold* = 5 ,*n-estimators* = 30 dan 40, dan *max-depth* = 20.

Setelah melakukan pengujian pertama, saya ingin mencoba melakukan pengujian kedua agar mendapatkan hasil persamaan *hamming loss* lebih baik lagi dengan mengganti pada bagian *preprocessing*, yang awalnya dengan *stemming* menjadi tanpa *stemming*. Pada percobaan tanpa *stemming*, saya tetap membandingkan kondisi 4-*Fold* dan 5-*Fold*. Selain itu tetap mempertahankan 2 parameter yaitu *n-estimators* dan *max-depth*. Adapun implementasi dari hasil percobaan kali ini dapat di lihat pada tabel 11 dan tabel 12 yaitu sebagai berikut.

**Tabel 11. Hasil *Hamming Loss* tanpa *Stemming* pada kondisi 4-*Fold***

| n-estimators | Max-depth | | | | | Mean |
|---|---|---|---|---|---|---|
| | 5 | 10 | 15 | 20 | 25 | |
| **10** | 0.13533 | 0.15600 | 0.15200 | 0.15000 | 0.15333 | **0.14933** |
| **20** | 0.14133 | 0.14867 | 0.15000 | 0.14467 | 0.14867 | **0.14667** |
| **30** | 0.14067 | 0.14400 | 0.14333 | 0.14267 | 0.14600 | **0.14333** |
| **40** | 0.14133 | 0.14333 | 0.14400 | 0.14133 | 0.14200 | **0.14240** |
| **50** | 0.14267 | 0.14533 | 0.14333 | 0.14333 | 0.14600 | **0.14413** |
| **60** | 0.14267 | 0.13867 | 0.14400 | 0.14133 | 0.14333 | **0.14200** |
| **70** | 0.13933 | 0.13733 | 0.14067 | 0.14200 | 0.14200 | **0.14027** |
| **80** | 0.13800 | 0.13667 | 0.13933 | 0.14267 | 0.14333 | **0.14000** |
| **90** | 0.13733 | 0.13533 | 0.14267 | 0.13933 | 0.14400 | **0.13973** |
| **100** | 0.13867 | **0.13467** | 0.14467 | 0.14067 | 0.14467 | **0.14067** |
| ***Mean*** | **0.13973** | **0.14200** | **0.14440** | **0.14280** | **0.14533** | |

**Tabel 12. Hasil *Hamming Loss* tanpa *Stemming* pada kondisi 5-*Fold***

| n-estimators | Max-depth | | | | | Mean |
|---|---|---|---|---|---|---|
| | 5 | 10 | 15 | 20 | 25 | |
| **10** | 0.13600 | 0.13867 | 0.13733 | 0.13533 | 0.13533 | **0.13653** |
| **20** | 0.14467 | 0.14000 | 0.13867 | 0.14000 | 0.14000 | **0.14067** |
| **30** | 0.14200 | 0.13533 | 0.13533 | 0.13867 | 0.14200 | **0.13867** |
| **40** | 0.13933 | 0.13067 | 0.13200 | 0.13400 | 0.13800 | **0.13480** |
| **50** | 0.14067 | 0.13133 | 0.13200 | 0.13533 | 0.13200 | **0.13427** |
| **60** | 0.13867 | 0.13200 | 0.13200 | 0.13467 | 0.13600 | **0.13467** |
| **70** | 0.14133 | 0.13133 | 0.13133 | 0.13400 | 0.13400 | **0.13440** |
| **80** | 0.13867 | 0.13067 | 0.13200 | 0.13133 | 0.13667 | **0.13387** |
| **90** | 0.13667 | 0.13067 | 0.13133 | **0.12933** | 0.13467 | **0.13253** |
| **100** | 0.13600 | 0.13000 | 0.13133 | 0.13000 | 0.13667 | **0.13280** |
| ***Mean*** | **0.13940** | **0.13307** | **0.13333** | **0.13427** | **0.13653** | |

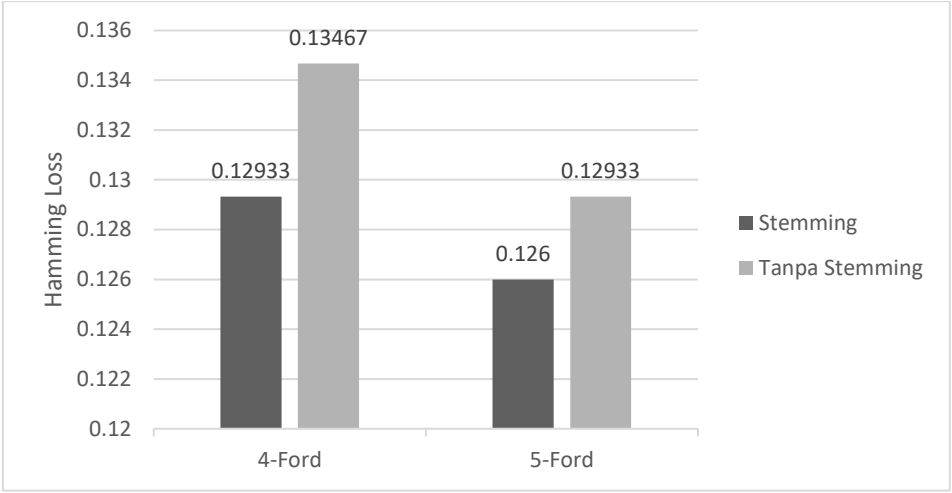
Pada kedua percobaan yaitu pada tabel 11 dengan kondisi 4-*Fold* dan pada tabel 12 dengan kondisi 5-*Fold* dilakukan perhitungan tanpa menggunakan proses *stemming* pada *preprocessing*. Pada percobaan tersebut diperoleh hasil dari persamaan *hamming loss*, contohnya saat kondisi 4-*Ford* lalu nilai *n-estimators* = 10 dan nilai *max-depth* = 5 maka diperoleh hasil persamaan *hamming loss* = 0,13533. Tujuan dari percobaan ini untuk menguji hasil terbaik dari kondisi 4-*Fold* dan 5-*Fold* tanpa melakukan proses stemming pada preprocessing dan mengubah nilai parameter pada *random forest* yaitu *n-estimators* dan *max-depth*.

Pada percobaan tabel 11 diperoleh hasil terbaik dari persamaan *hamming loss* yaitu 0,13467 dimana nilai *k-Fold* = 4, *n-estimators* = 100, dan *max-depth* = 10. Sedangkan pada tabel 12 diperoleh hasil terbaik dari persamaan *hamming loss* yaitu 0,12933 dimana nilai *k-Fold* = 5 ,*n-estimators* = 90, dan *max-depth* = 20. Jika dibandingkan maka persamaan *hamming loss* terbaik dari tabel 11 dengan 4-*Fold* dan tabel 12 dengan 5-*Fold* maka nilai

persamaan *hamming losss* terbaik pada kondisi tanpa *stemming* yaitu 0,12933 dimana *k-Fold* = 4, *n-estimators* = 100, dan *max-depth* = 10.

4.2 Analisis Hasil Pengujian

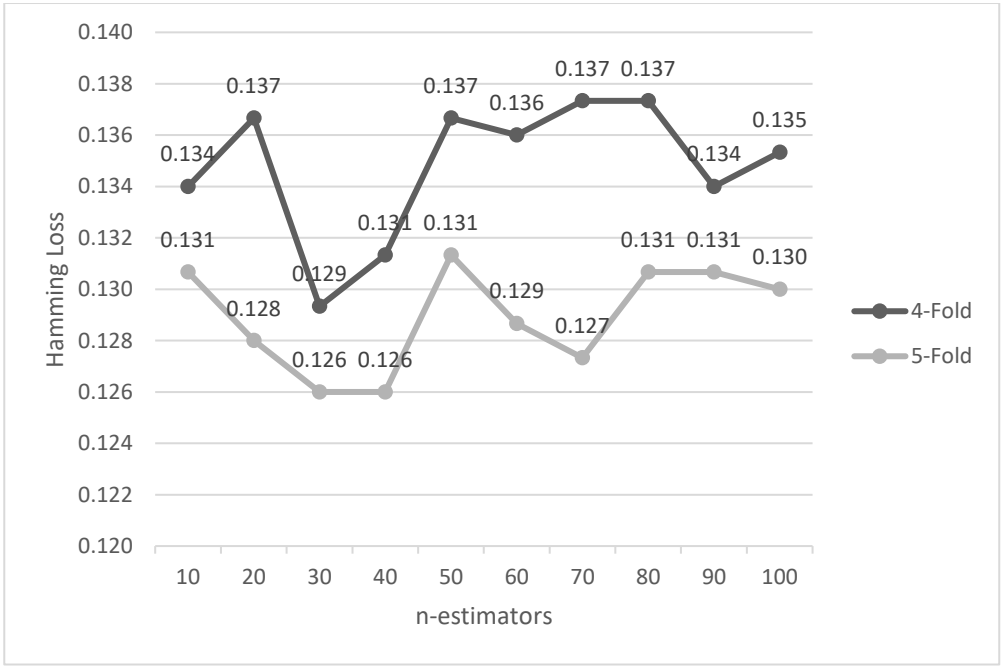
Pada percobaan pertama dan kedua yaitu menggunakan proses *stemming* dan tanpa *stemming*. Saya mencoba untuk membandingkan kedua pengujian tersebut dimana memiliki nilai *k-Fold* yang sama. Tujuan dilakukannya hal ini, agar dapat membandingkan persamaan terbaik antara *stemming* dan tanpa *stemming* dimana diambil hasil terbaik dari proses tersebut. Berikut perbandingan dari hasil pengujian yaitu pada gambar 5 sebagai berikut.




**Gambar 5. Perbandingan persamaan *Hamming Loss Stemming* dan tanpa *Stemming***

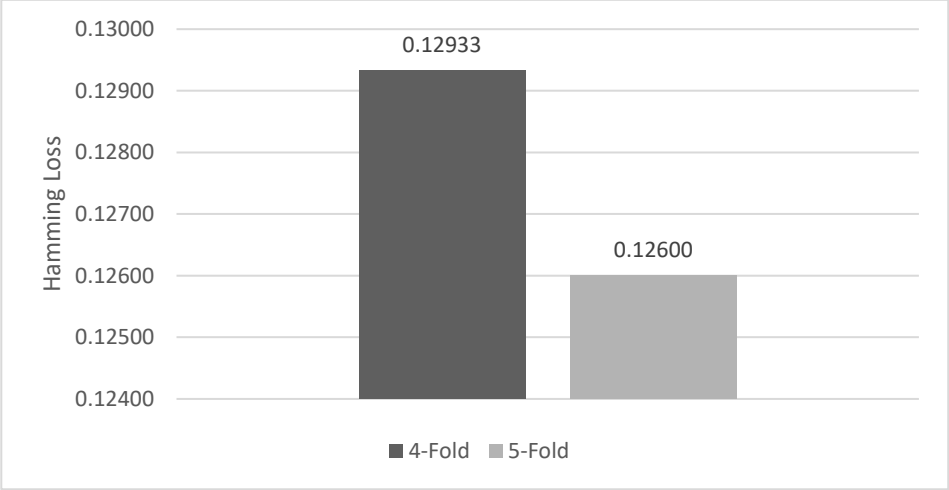
Pada gambar 5, dikedua kondisi yaitu 4-*Fold* dan 5-*Fold* diperoleh proses dengan *Stemming* lebih baik dibandingkan dengan proses tanpa *stemming*. Hal ini dapat terjadi dikarenakan, adanya pengubahan menjadi kata dasar pada proses *stemming*, yang menyebabkan proses *stemming* lebih baik dibandingkan proses tanpa *stemming*.

Pada percobaan sebelumnya, diperoleh proses *stemming* yang lebih baik. Pada tabel 9 dilakukan dengan kondisi 4-*Ford* dan pada tabel 10 dilakukan dengan kondisi 5-*Ford*. Pada kedua percobaan tersebut menggunakan proses *Stemming* pada *preprocessing*. Pada tabel tabel 9 diperoleh persamaan *hamming loss* terbaik = 0,12933 dimana nilai *k-Fold* = 4, *n-estimators* = 30, dan *max-depth* = 15 dan 20. Sedangkan pada tabel 10 diperoleh persamaan *hamming loss* terbaik = 0,12600 dimana nilai *k-Fold* = 5 ,*n-estimators* = 30 dan 40, dan *max-depth* = 20. Dari hal tersebut, saya mencoba membandingkan kedua kondisi tersebut dalam bentuk grafik pada gambar 6 yaitu sebagai berikut.




**Gambar 6. Perbandingan persamaan *Hamming Loss* dengan *Stemming* dan max-depth = 20**

Pada gambar 6 dapat dilihat hasil perbandingan 4-*Fold* dan 5-*Fold* dalam kondisi *stemming* dan *max-depth* = 20. Pada gambar tersebut, hasil persamaan *hamming loss* terbaik yaitu 5-*Fold*, dimana setiap kondisi n = 10 sampai 100, nilai 5-*Fold* lebih baik dari 4-*Fold*. Hal bisa saja terjadi dikarenakan 5-*Fold* sesuai dengan *rule of thumb* (aturan umum) dengan proporsi *test set* sebesar 20% dan *train set* sebesar 80%, dikarenakan konsep dari *k-Fold* yaitu semua *dataset* harus diuji menjadi *test set* dan *train set*. Pada saat, kondis 5-*Fold* maka data sebanyak *k* dan mengulangi (men-iterasi) eksperimennya sebanyak *k* juga, jadi iterasi yang dilakuakan sebanyak 5 dengan *test set* = 100 dan *train set* = 400 yang akan diganti berulang dengan angka yang berbeda sebanyak 5 kali dibnadingkan dengan 4-*Fold* yang iterasinya lebih sedikit. Jika diambil masing-masing nilai terbaik dari kondisi 4-*Fold* dan 5-*Fold*, maka dapat dibandingkan pada gambar 7 sebagai berikut.




**Gambar 7. Perbandingan nilai *hamming loss stemming* dan tanpa *stemming***

Pada skenario perbandingan 4-*Fold* dan 5-*Fold* pada gambar 7 diperoleh hasil persamaan *hamming loss* terbaik yaitu 0,126 dimana nilai *k-Fold* = 5, *n-estimators* = 30, dan *max-depth* = 20. Hasil tersebut diperoleh, berawal dari menghitung nilai persamaan *hamming loss* untuk 4-*Fold* dan 5-*Fold* dengan menggunakan 2 parameter pada *random forest classifer* yaitu *n-estimators* dan *max-depth*. Selain itu, saya mencoba juga membandingkan dengan menghapus kondisi stemming pada *preprocessing* untuk membuktikan apakah proses tanpa *stemming* menghasilkan nilai yang lebih baik dibandingkan proses dengan *stemming*. Alhasil yang diperoleh proses *stemming* masih lebih baik dibandingkan tanpa *stemming*(dapat dilihat pada gambar 5). Setelah itu, jika melihat tabel 9 dan 10 dapat dilihat hasil terbaik berada pada kondisi *max-depth* = 20, karena hal itu saya mencoba membandingkan kondisi *n-estimators* untuk melihat performansi yang lebih baik antar 4-*Fold* dan 5-*Fold* (dapat dilihat pada gambar 6). Dari hal itu, performansi terbaik adalah disaat kondisi 5-*Fold* dan jika diambil nilai terbaik dari masing-masing *k-Fold* tersebut (pada gambar 7) maka kondisi 5-*Fold* lebih baik.

Jadi berdasarkan hal tersebut, didapat persamaan *hamming loss* terbaik adalah 0,126 dimana nilai tersebut dapat dikatakan berhasil jika dibandingkan pada penelitian sebelumnya.

## 5. Kesimpulan

Dari hasil pengujian yang dilakukan dari 500 *dataset* berita dihasilkan dengan beberapa tahap dapat disimpulkan bahwa mengklasifikasikan berita menggunakan metode *random forest* tanpa menggunakan proses s*temming* pada *preprocessing* menghasilkan nilai *hamming loss* yaitu 0,126. Dapat disimpulkan, hasil dari kategorisasi berita menggunakan algoritma *random forest* lebih baik. Karena, jika dibandingkan pada penelitian sebelumnya yang menghasilkan nilai persamaan *hamming loss* sebesar 0,1472.

Adapun saran dari peneliti untuk penelitian selanjutnya yaitu menambahkan jumlah *dataset* yang digunakan, karena dengan penambahan jumlah *dataset* diharapkan mampu menambahkan jumlah dan keragaman informasi sehingga dapat menambah informasi dari metode *random forest* dan meningkatkan performa sistem yang dibangun.

## Daftar Pustaka

[1] D. Fu, B. Zhou and J. Hu, "Improving svm based multi-label classification by using label relationship. In Neural Networks (IJCNN)," *International Joint Conference on IEEE,* pp. 1-6. , 2015.

[2] M. Ben-Dov and F. R. , "Text Mining and Information Extraction, Chapter 38.," *Data Mining and Knowledge Discovery Handbook,* pp. 801-831, 2001.

[3] F. Sebastiani, Machine learning in automated text categorization, ACM Computing Surveys, Vol. 34 (1), 1-47., 2002.

[4] Nurfikri, F. Salman and M. S. Mubarok, "News Topic Classification Using Mutual and Bayesian Network," *In 2018 6th International Conference on Information and Communication Technology (ICoICT),* pp. 162-166, 20018.

[5] Adiwijaya, M. N. Aulia, M. S. Mubarok, W. U. Novia and F. Nhita, "A comparative study of MFCC-KNN and LPC-KNN for hijaiyyah letters pronounciation classification system.," *In 2017 5th International Conference on Information and Communication Technology (ICoIC7). IEEE.,* pp. 1-5, 2017.

[6] R. A. Pane, M. S. Mubarok, N. S. Huda and A. Adiwijaya, "A multi-lable classification on topics of quranic verses in english translation using multinomial naive bayes," *In Information and Communication Technology(ICoICT. IEEE), 2018 6th International Conference on. IEEE,* 2018.

[7] A. I. Pratiwi and Adiwijaya, "On the Feature Selection and Classification Based on Information Gain for Document Sentiment Analysis," *Applied Computational Intelligence and Soft Computing, 2018,* 2018.

[8] S. N. Asiyah and K. Fithriasari, "Klasifikasi berita online menggunakan metode support vector machine dan k-nearest neighbor," *Jurnal Sains dan Seni ITS,* p. 5(2), 2016.

[9] A. K. Usyal and S. Gunal, "The impact of preprocessing on text classification," *Information Processing and Management, vol.50,* pp. pp. 104-112, 2014.

[10] J. Govaerts, Regular Expressions: The Complete Tutorial, http://regular-expressions.info/print.html, 2007.

[11] F. Bastiani, "Machine learning in automated text categorization," *ACM Computing Surveys, 34,* pp. 1-47, 2002.

[12] M. Z. Naf'an, A. A. Bimantara, A. Larasati, E. M. Risondang and N. A. S. Nugraha, "Sentiment Analysis of Cyberbullying on Instagram User Comments," *Journal of Data Science and Its Applications, 2(1),* pp. 88-98, 2019.

[13] M. Y. A. Bakar, Adiwijaya and S. Al Faraby, "Multi-Label Topic Classification of Hadith of Bukhari (Indonesian Language Translation) Using Information Gain and Backpropagation Neural Network.," *In 2018 International Conference on Asian Language Processing (IALP). IEEE,* pp. 344-350, 2018.

[14] P. Rafaeilzadeh, L. Tang and H. Liu, Cross-validation, DOI: https://doi.org/10.1007/978-0-387-39940-9_565, 2008.

[15] R. A. Aziz, M. S. Mubarok and Adiwijaya, "Klasifikasi Topik pada Lirik Lagu dengan Metode Multinomial Naive Bayes," *In Indonesia Symposium on Computing (IndoSC) 2016,* 2016.

[16] V. Botta, A walk into random forests Adaptation and application, Faculty of Applied Sciences Department of Electrical Engineering and Computer Science, 2013.

[17] M. D. Purbolaksono, K. C. Widiastuti, Adiwijaya, M. S. Mubarok and F. A. Ma'ruf, "Implementation of mutual information and bayes theorem for classification microarray data," *In Journal of Physics: Conference Series (Vol. 971, No. 1, p. 012011). IOP Publishing,* 2018.

[18] S. Al Faraby, E. R. R. Jasin and A. Kusumaningrum, "Classification of hadith into positive suggestion, negative suggestion, and information," *In Journal of Physics: Conference Series (Vol. 971, No. 1, p. 012046). IOP Publishing.,* 2018.

**Lampiran**

Lampiran dapat berupa detil data dan contoh lebih lengkapnya, data-data pendukung, detail hasil pengujian, analisis hasil pengujian, detail hasil survey, surat pernyataan dari tempat studi kasus, screenshot tampilan sistem, hasil kuesioner dan lain-lain.