BAB 2
TINJAUAN PUSTAKA

2.1 Analisis Sentimen

Analisis sentimen atau yang juga dikenal sebagai opinion mining adalah sebuah bidang studi yang berguna untuk menganalisis pendapat, penilaian, sikap, dan perasaan seseorang tentang suatu produk, layanan, masalah, isu, dan sebagainya[1]. Selain itu, analisis sentimen juga dapat digunakan untuk melihat pendapat atau kecenderungan beropini negatif atau positif. Data yang digunakan dalam analisis sentimen dapat diperoleh melalui media sosial, maupun ulasan dalam suatu aplikasi yang biasanya terdapat penilaian terhadap suatu brand, tokoh, produk, kebijakan, serta hal-hal lain yang dicurahkan ke dalam sebagai opini sehingga dapat dijadikan masukan untuk melakukan perbaikan pada suatu perusahaan maupun tokoh dan sebagainya. Analisis sentimen juga dapat digunakan dalam mengelompokan emosi seperti senang, sedih, marah, terkejut.

2.2 Kata Tidak Baku

Kata tidak baku merupakan kata yang biasa digunakan dalam keadaan yang informal atau tidak resmi. Beberapa kata dikategorikan sebagai kata tidak baku seperti kesalahan dalam kaidah penulisan, ejaan kata, dan singkatan kata. Dengan adanya kata tidak baku, maka perbaikan untuk kata tidak baku menjadi kata baku dapat dilakukan. Perbaikan kata tidak baku merupakan normalisasi pada Bahasa yang digunakan dalam mengubah kata tidak baku menjadi kata baku yang sesuai dengan Kamus Besar Bahasa Indonesia (KBBI) [9].

Pada perbaikan kata tidak baku, ada beragam kata yang dianggap tidak baku, diantaranya penulisan *slang word* (ex : umak = kamu), singkatan kata (ex : dmn = dimana), kesalahan kaidah penulisan (ex : resiko = risiko) , kesalahan dalam ejaan kata (ex : pimtar = pintar). Penelitian ini akan memperbaiki kata tidak baku dan kesalahan dalam ejaan kata atau *typo correction*.

2.3 *Typo Correction*

Saat menguatarakan opini dalam media sosial sering kali pengguna melakukan kesalahan dalam ejaan atau penulisan. Dengan adanya kesalahan ejaan tersebut dapat mengakibatkan kata yang ditulis tidak sesuai atau tidak terdaftar pada Kamus Besar Bahasa Indonesia (KBBI). Kata yang tidak sesuai ini akan diubah menjadi kata yang sesuai atau menjadi kata yang terdapat dalam Kamus Besar Bahasa Indonesia (KBBI) dengan melakukan *Typo correction.*

Typo correction adalah sebuah proses mengenali, memeriksa dan mendeteksi kata-kata yang terdapat kesalahan dalam ejaan. Penelitian ini mengatasi *non-real word error,* yaitu kata yang dapat diperbaiki hanya kata yang tidak memiliki arti di dalam kamus [10]. Contoh, kurangnya huruf seperti ‘aym’ yang seharusnya ayam, kesalahan penulisan huruf ‘nuta’ yang seharusnya ‘buta’ dan adanya huruf berlebih seperti ‘bajuu’ yang seharusnya ‘baju’.

2.4 *Preprocessing*

Preprocessing adalah salah satu teknik pada data mining yang di dalamnya terdapat perubahan data mentah menjadi format yang lebih terstruktur dan mudah dipahami[11]. Tahapan ini diperlukan karena sebuah data yang diambil biasanya masih berupa data mentah yang memiliki simbol, dan kata-kata yang tidak diperlukan. Pada penelitian ini ada beberapa tahap *preprocessing* yang digunakan, diantaranya *case folding, cleaning, tokenizing, word normalization, convert negation, stemming,* dan *filtering*.

2.4.1 *Case Folding*

Case Folding adalah tahap dalam *preprocess* untuk mengubah karakter atau huruf menjadi huruf kecil (*lower case*)[12]. Tahap ini dilakukan karena dalam penulisan sebuah tweet selalu ada perbedaan bentuk huruf yang digunakan penulis. Contoh penulisan ‘Nama saya Budiono’ menjadi ‘nama saya budiono’.

2.4.2 *Cleaning*

Cleaning merupakan proses penghapusan format asal pada media sosial dan simbol yang terdapat pada dataset[13]. Tahap ini digunakan untuk mengurangi

noise, di mana pengubahan menjadi spasi dilakukan pada angka, tanda baca, serta simbol-simbol seperti '@' untuk *username*, *username, hastag* (#), alamat *website*, *mention, link, emoticon,* spasi berlebih, dan karakter ganda.

2.4.3 *Tokenizing*

Tokenize adalah tahap untuk membagi-bagi kalimat pada data tweet atau *dataset*, data yang awalnya berbentuk kalimat diubah menjadi bentuk kata perkata[14]. Pada tahap ini kalimat akan dipecah berdasarkan spasi menjadi kata-kata tunggal.

2.4.4 *Word Normalization*

Word Normalization atau normalisasi kata merupakan proses pengubahan kata-kata yang tidak baku atau tidak sesuai dengan KBBI (Kamus Besar Bahasa Indonesia) menjadi kata baku[15]. Pada tahapan ini akan dilakukan pemeriksaan kata apakah terdapat pada kamus atau tidak. Jika kata tidak baku ditemukan pada kamus kata tidak baku, maka kata tersebut akan diubah menjadi kata baku.

2.4.5 *Convert Negation*

Convert Negation merupakan tahap pengubahan kata atau konversi kata negasi yang ada dalam suatu kalimat. Kata negasi dapat membalikan atau merubah kata yang sebenarnya [16]. Selain itu kata negasi mempunyai pengaruh terhadap nilai analisis sentimen. Kata-kata negasi meliputi kata "tidak", "tak", "bukan", "kurang", "belum" dan "tanpa". Jika ditemukan, kata-kata ini akan digabungkan dengan kata setelahnya.

2.4.6 *Stemming*

Stemming merupakan tahap pengubahan kata dasar yang sesuai dengan kaidah pada Kamus Besar Bahasa Indonesia[17]. Pengubahan kata menjadi kata dasar dapat dilakukan dengan menghilangkan imbuhan. Contoh kata 'kewajiban' diubah menjadi kata dasar 'wajib'.

2.4.7 *Filtering*

Filtering merupakan tahap menghilangkan atau penghapusan kata-kata yang ada dalam jumlah banyak tetapi tidak mempunyai arti (*stopwords*)[17].

Contoh kata-kata yang tidak memiliki arti diantaranya adalah tetapi, atau, yang, di, ke, bila, dan lain sebagainya.

2.5 Twitter

Twitter merupakan media berbasis internet atau disebut sebagai sosial media yang dapat digunakan pengguna untuk mengirim pesan secara *realtime*, atau yang disebut sebagai tweet[18]. Twitter sering digunakan penggunanya dalam mengutarakan emosi tentang berbagai hal, baik berupa pujian maupun kritik. Selain itu, Twitter dapat dijadikan tempat untuk berbagi informasi, ataupun opini tentang berbagai hal. Opini tersebut dapat bersifat positif, maupun negatif.

2.6 *Levenshtein Distance*

Levenshtein Distance adalah sebuah metode yang dapat digunakan dalam mengatasi terjadinya kesalahan ejaan kata [3]. Fungsi metode *Levenshtein Distance* yaitu untuk mengukur jarak antara dua buah string melalui penambahan, pengubahan dan penghapusan karakter[19].

Pada algoritma *Levenshtein Distance* ada 3 buah operasi utama yang dilakukan, yaitu :

1. Penghapusan Karakter

   Operasi penghapusan karakter merupakan sebuah operasi yang didalamnya terdapat penghapusan untuk menghilangkan sebuah karakter pada suatu *string*. Contohnya ‘kamuu’, karakter terakhir perlu dihilangkan sehingga *string* menjadi ‘kamu’. Pada operasi ini huruf ‘u’ yang berlebih pada kata ‘kamu’ dihapuskan atau dihilangkan.
2. Pengubahan Karakter

   Operasi pengubahan karakter merupakan sebuah operasi yang menukar satu karakter dengan karakter lain. Contohnya penulisan ‘psnjang’ menjadi ‘panjang’. Dalam kasus ini karakter ‘s’ diganti dengan ‘a’.
3. Penambahan Karakter

   Operasi penambahan karakter merupakan sebuah operasi yang berarti menambahkan sebuah karakter atau huruf kedalam kata atau *string*. Contohnya

‘sian’ menjadi ‘siang’, dilakukan penambahan karakter ‘g’ di akhir kata. Penambahan ini juga dapat dilakukan di awal, di tengah, maupun di akhir kata.

Perbandingan akan dilakukan pada kata sumber terhadap setiap kata yang ada di dalam kamus lalu nilai *edit distance* akan didapatkan dari nilai yang berada pada pojok kanan bawah yang menggambarkan jumlah perbedaan dari kedua string, seperti pada contoh perhitungan yang ditunjukkan pada gambar 2.1.

		Target						
		N	A	S	I	H	A	T
	0	1	2	3	4	5	6	7
N	1	0	1	2	3	4	5	6
A	2	1	0	1	2	3	4	5
S	3	2	1	0	1	2	3	4
E	4	3	2	1	1	2	3	4
H	5	4	3	2	2	1	2	3
A	6	5	4	3	3	2	1	2
T	7	6	5	4	4	3	2	1

Sumber

Gambar 2.1 Matriks Perhitungan *Edit Distance*

Gambar 2.1 menunjukan perhitungan *edit distance* menggunakan 2 *string* yaitu nasehat dan nasihat, sehingga menghasilkan nilai *edit distance* sebesar 1. Perubahan kata akan dilakukan dengan kata yang memiliki nilai *edit distance* terkecil diantara semua kata yang telah dibandingkan.

2.7 Klasifikasi

Klasifikasi adalah pengelompokan data berdasarkan karakteristik ke kelas yang sesuai [20]. Pengklasifikasian ini dibuat dari kumpulan data dengan kelas

yang telah diketahui atau ditentukan. Klasifikasi termasuk ke dalam *supervised learning* karena pada pengelompokannya disediakan kategori pola dalam data latih.

Pengklasifikasian akan melalui 2 proses yaitu *training* dan *testing*. *Training* dilakukan untuk menghasilkan data latih, sedangkan *testing* yaitu proses yang dilakukan terhadap data uji.

2.8 Support Vector Machine

Support Vector Machine (SVM) merupakan metode klasifikasi dalam *supervised learning*. Metode ini sering dipakai untuk klasifikasi otomatis, parameter yang didapat dari pelatihan metode SVM dijamin merupakan parameter optimal [21]. Tujuan klasifikasi Support Vector Machine (SVM) adalah menciptakan komputasi yang efisien, memisahkan *hyperplanes* pada fitur dimensi agar mengoptimalkan batas [22]. *Hyperplane* yang terdapat pada SVM merupakan fungsi linear. Data latih (xi, yi), dimana xi merupakan atribut kelas ke-i dan yi menyatakan label kelas. Data xi terbagi kedalam kelas positif (+1) dan kelas negatif (-1) yang ditunjukan pada persamaan 2.1 dan 2.2.

$w \times xi + b < 0$ untuk yi = -1 (2.1)

$w \times xi + b > 0$ untuk yi = +1 (2.2)

Keterangan :

w = nilai bidang normal

b = posisi bidang terhadap pusat koordinat

xi = data input

yi = label

Dengan memaksimalkan jarak pembatas antara 2 kelas dan titik terdekat dapat dilakukan untuk menemukan margin terbesar yang dirumuskan sebagai *quadratic programming problem*. Dengan mencari titik minimal pada persamaan 2.3, dan memperhatikan persamaan 2.4.

$$min\ \tau(w)\ =\ \frac{1}{2}\|w\|^2 \quad (2.3)$$

$$yi(w\ \times\ xi\ +\ b)\ -\ 1\ \geq\ 0,\ (i\ =\ 1,2,\ldots,n) \quad (2.4)$$

Persamaan 2.3 dapat dipermudah dengan menggunakan fungsi Lagrangian yang ditunjukan pada persamaan 2.5.

$$L(w,b,a)\ =\ \frac{1}{2}\|w\|^2\ -\sum_{i=1}^{n}\alpha_i\ [\ y_i\ ((w\ \times\ x_i\ +\ b)\ -\ 1)] \quad (2.5)$$

Dimana α_i merupakan *lagrange multiplier* bernilai $\geq$ 0. Dengan meminimalkan L terhadap w, b, dan a dapat menghasilkan nilai optimal pada persamaan 2.5. persamaan tersebut ditunjukan pada persamaan 2.6, 2.7 dan 2.8.

$$\frac{\partial L}{\partial w} = w\ -\sum_{i=1}^{n}\alpha_i y_i x_i =\ 0 \quad (2.6)$$

$$\frac{\partial L}{\partial b} = w\ -\sum_{i=1}^{n}\alpha_i y_i =\ 0 \quad (2.7)$$

$$\frac{\partial L}{\partial b} = w\ -\sum_{i=1}^{n}\alpha_i y_i(w^\tau\times\ x_i\ +\ b)\ -\ \sum_{i=1}^{n}\alpha_i =\ 0 \quad (2.8)$$

Masalah lagrangian dijelaskan pada persamaan 2.9 untuk pengklasifikasian.

$$min\ L(w,b,a)\ =\ \frac{1}{2}\|w\|^2\ -\sum_{i=1}^{n}\alpha_i y_i(w^\tau\times\ x_i\ +\ b)\ -\ \sum_{i=1}^{n}\alpha_i \quad (2.9)$$

Persamaan 2.10 dapat digunakan untuk memaksimalkan L terhadap α_i.

$$Max\sum_{i=1}^{n}\alpha_i\ -\ \frac{1}{2}\sum_{i=1,j=1}^{n}\alpha_i\alpha_j y_i y_j^\tau x_i x_j^\tau \quad (2.10)$$

Setelah mendapatkan nilai dari pembobotan kata dengan TF-IDF dan inisialisasi kelas, dapat dilakukan pencarian untuk nilai yi dan xi. Label akan didapat dari data kelas, dan hasil pembobotan diubah menjadi bentuk data SVM. Data perlu diubah menjadi *support vector* = $\binom{x}{y}$, lalu nilai tersebut dimasukkan ke dalam persamaan kernel trick phi 2.11.

$$\phi\begin{bmatrix}x\\y\end{bmatrix} = \begin{cases}\sqrt{x\frac{2}{n} + y\frac{2}{n}} > 2\,maka & \begin{bmatrix}\sqrt{x\frac{2}{n}+y\frac{2}{n}} - x + |x-y|\\ \sqrt{x\frac{2}{n}+y\frac{2}{n}} - y + |x-y|\end{bmatrix}\\ \sqrt{x\frac{2}{n} + y\frac{2}{n}} \leq 2\,maka & \begin{bmatrix}x\\y\end{bmatrix}\end{cases} \quad (2.11)$$

Nilai x diperoleh dari persamaan 2.12 dan nilai y diperoleh dari persamaan 2.13.

$$\sum_{i=1,j=1}^{n} x_i x_j^{\tau}, (i,j = 1,2,\ldots,n) \quad (2.12)$$

$$\sum_{i=1,j=1}^{n} y_i y_j^{\tau}, (i,j = 1,2,\ldots,n) \quad (2.13)$$

Selanjutnya parameter α_i dicari dengan menghitung nilai fungsi setiap data dengan persamaan 2.14, lalu parameter α_i dimasukkan ke dalam persamaan 2.15.

$$\sum_{i=1,j=1}^{n} \alpha_i S_i^{\tau} S_j = y_j \quad (2.14)$$

$$\widehat{W} = \sum_{i=1}^{n} \alpha_i S_i \quad (2.15)$$

Setelah mendapatkan hasil dari persamaan 2.15, digunakan persamaan 2.16 untuk menghasilkan nilai w dan b.

$$y = wx + b \quad (2.16)$$

Nilai *hyperplane* yang telah didapatkan digunakan untuk mengklasifikasikan kelas.

2.8.1 Kernel Trick

Dengan menggunakan *kernel trick*, hanya perlu diketahui fungsi kernel yang digunakan dalam menentukan *support vector*, sehingga tidak perlu wujud dari fungsi non-linier digunakan dalam mengetahui fungsi kernel yang SVM dapat diperluas dengan tujuan untuk mendapatkan batas keputusan non-linier ϕ [23]. Untuk itu, SVM memiliki kernel *trick* yang dapat membantu mengatasi perubahan data menjadi ruang non-linier. Secara umum, beberapa fungsi kernel pada SVM yang dapat digunakan sebagai berikut[23]:

1. Kernel Linier

$$K(x, x_k) = x_k^T x \quad (2.17)$$

2. Kernel Polynomial

$$K(x, x_k) = (x_k^T x + 1)^d \quad (2.18)$$

3. Kernel Gaussian (radial basis function, RBF)

$$K(x, x_k) = exp\{-\|x - x_k\|_2^2 / \sigma^2\} \quad (2.19)$$

4. Kernel Sigmoid

$$K(x, x_k) = \tanh[k\, x_k^T\, x + \theta] \quad (2.20)$$

2.9 Pembobotan Kata TF-IDF

Data yang sudah melalui *preprocessing* perlu diubah menjadi bentuk numerik, sehingga perlu dilakukannya pembobotan kata[24]. TF-IDF (*Term frequency–inverse document frequency*) adalah proses untuk mentransformasi data tekstual menjadi numerik untuk pembobotan setiap kata. TF-IDF juga merupakan stastistik numerik untuk mencerminkan pentingnya sebuah kata dalam dokumen [25]. Metode ini menggunakan frekuensi kemunculan kata di dalam dokumen. Konsep metode ini mempunyai kelemahan yaitu kata dianggap selalu setara yang menyebabkan relevansi kata menjadi tinggi ketika sering muncul dalam sebuah dokumen. Dalam perhitungannya nilai *term frequency (tf)* dihitung berdasarkan kemunculan kata tersebut dalam dokumen. Nilai *tf* dihitung dengan rumus *weighting term frequency* yang ditunjukan pada persamaan 2.21.

$$W_t\, f_t, d = \begin{cases} 1 + log_{10}\, tf_t, d\, , if\ tf_t, d\ > 0 \\ 0,\ otherwise \end{cases} \quad (2.21)$$

Dalam sebuah dokumen umumnya ada beberapa kata yang tidak penting, maka dari itu diperlukan pembobotan *document frequency* untuk menghitung jumlah atau frekuensi dokumen yang didalamnya terdapat term atau kata. Dengan nilai term yang ditemukan pada setiap dokumen, dilakukan kebalikan dari proses *document frequency* yaitu proses *inverse document frequency*. Proses ini memunculkan bobot yang tinggi terhadap kata yang jarang muncul pada setiap dokumen. Dengan rumus persamaan 2.22.

$$idf_t\ =\ log_{10}(\frac{N}{df_t}) \quad (2.22)$$

Dimana :

idf_t = bobot *inverse* nilai df.

N = Jumlah dokumen.

Df_t = Jumlah dokumen yang mengandung term atau kata.

Pembobotan TF-IDF merupakan hasil perkalian *term frequency* dengan *inverse document frequency*, sehingga didapatkan rumus yang ditunjukan pada persamaan 2.23.

$$w_{t,d} = TF\ x\ IDF = TF\ x \log_e \frac{|D|}{DF} \quad (2.23)$$

Keterangan :

$W_{t,d}$ = Pembobotan TF-IDF.

TF = Bobot kata pada setiap dokumen (*term frequency).*

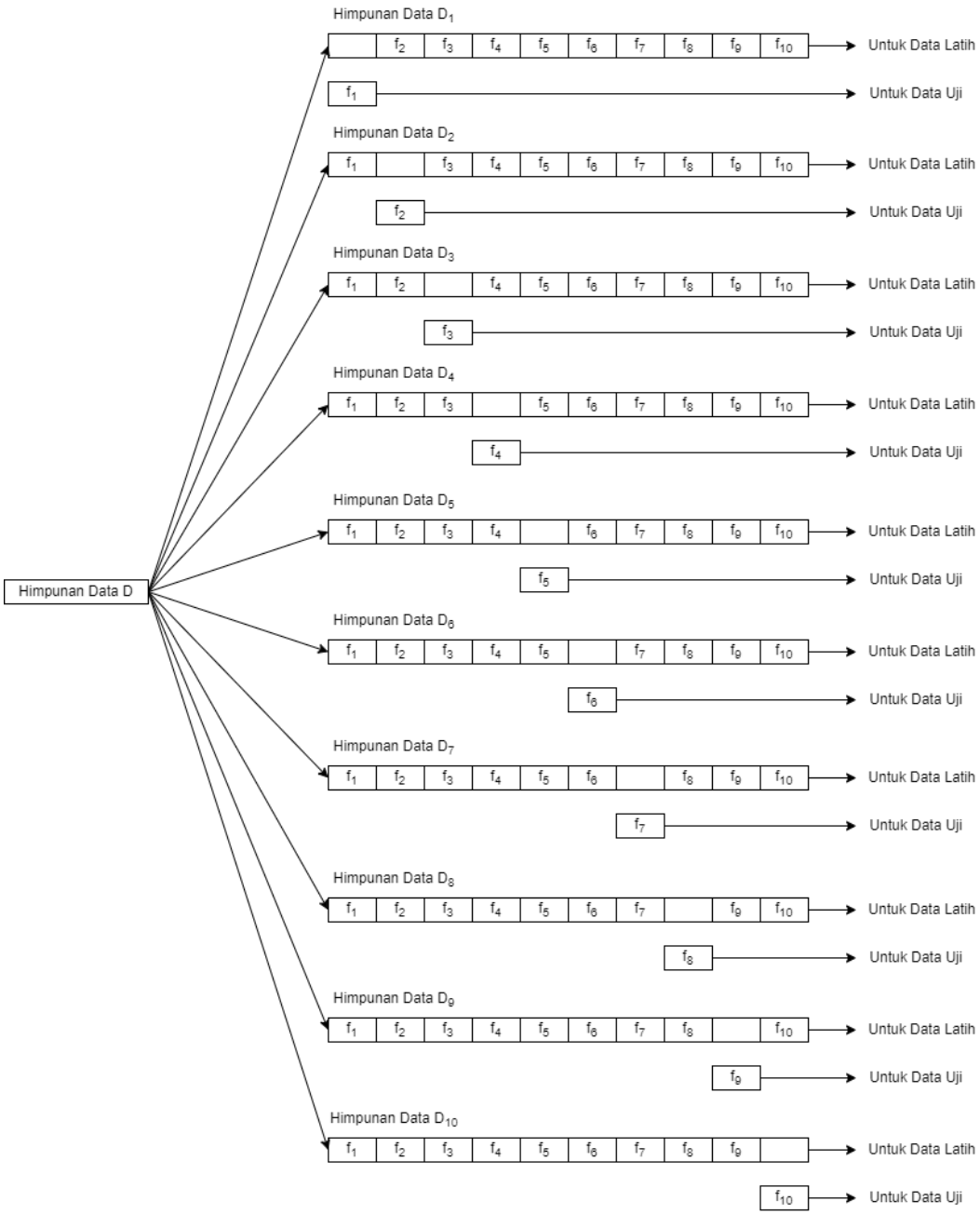
IDF = bobot inverse nilai *df.*

2.10 Metode Pengujian

Dalam penelitian ini metode pengujian yang digunakan yaitu *Confusion Matrix* dan *K-Fold Cross Validation.*

2.10.1 *K-Fold Cross Validation*

K-fold Cross Validation adalah sebuah metode proses validasi untuk memperkirakan kinerja dari model pembelajaran mesin atau *machine learning* [5]. Secara umum, *K-fold Cross Validation* digunakan untuk memperkirakan akurasi karena biasnya yang relatif rendah [26]. *K-fold Cross Validation* digunakan juga sebagai penghilang *noise* kata untuk meningkatkan akurasi dengan cara mengerjakan perulangan dengan masukan atribut acak. Diawali dengan pembagian data sejumlah *n-fold*. *Fold* ke-1 berarti data uji berada pada bagian ke-1, dan sisanya berupa data latih, *Fold* ke-2 berarti data uji berada pada bagian ke-2, dan sisanya berupa data latih, begitu seterusnya hingga *fold* ke-n. Berikut gambar dari metode *K-fold Cross Validation* dengan k = 10, ditunjukan pada gambar 2.2.



Gambar 2.2 Metode *K-Fold Cross Validation*

Metode *K-fold Cross Validation* dapat digunakan dalam mengukur kualitas dari berbagai model klasifikasi, membandingkan metode klasifikasi, dan memilih model terbaik diantara model yang dibangun [23].

2.10.2 *Confusion Matrix*

Confusion Matrix merupakan sebuah tabel yang berisi klasifikasi jumlah *testing data* yang benar atau sesuai dan jumlah *testing data* yang salah [27]. Metode

ini digunakan untuk mengetahui kinerja dari sebuah sistem yang akan dievaluasi. *Confusion Matrix* berguna dalam menganalisis kualitas suatu *classifier* mengenali tuple [23].

Tabel 2. 1 Confusion Matrix

		Kelas Prediksi	
		1	0
Kelas Sebenarnya	1	TP	FN
	0	FP	TN

Keterangan :

1. False Positive (FP) merupakan jumlah dokumen dari kelas 0 yang salah diklasifikasikan sebagai kelas 1.
2. False Negative (FN) merupakan jumlah dokumen dari kelas 1 yang salah diklasifikasikan sebagai kelas 0.
3. True Positive (TP) merupakan jumlah dokumen dari kelas 1 yang benar diklasifikasikan sebagai kelas 1.
4. True Negative (TN) merupakan jumlah kelas dari kelas 0 yang benar diklasifikasikan sebagai kelas 0.

Perhitungan *accuracy, recall,* dan *precision* dapat dilakukan dengan rumus *Confusion* Matrix yang ditunjukan pada persamaan 2.24, 2.25, 2.26.

$$Accuracy = \frac{\text{Prediksi Data Benar}}{\text{Total Data}} = \frac{TP + TN}{Total} \quad (2.24)$$

$$Recall = \frac{relevant\ item\ retrieved}{retrieved\ item} = \frac{TP}{TP + FN} \quad (2.25)$$

$$Precision = \frac{relevant\ item\ retrieved}{retrieved\ item} = \frac{TP}{TP + FP} \quad (2.26)$$

Accuracy digunakan dalam menyatakan persentase jumlah tuple pada data uji yang telah diklasifikasikan dengan benar. *Recall* merupakan ukuran kelengkapan dari besar persentase tuple positif yang dilabeli positif. *Precision* merupakan ukuran kepastian dari besar persentase tuple yang benar dilabeli positif.