BAB 2
LANDASAN TEORI

2.1 Analisis Sentimen

Analisis sentimen adalah bidang studi yang mebahas mengenai analisis pendapat orang, penilaian, sikap, dan emosi terhadap entitas tertentu; seperti: produk, layanan, organisasi, individu, masalah peristiwa, topik[7]. Analisis sentimen berfokus kepada opini publik yang mengekspresikan apakah sentimen tersebut positif, negative, atau netral. Secara umu, analisis sentiment terbagi menjadi 3 tingkat atau level, yaitu:

1. Level pada Dokumen.
   Pada level dokumen, akan diklasifikasi, apakah seluruh pendapat yang ada di dalam dokumen bersifat negatif atau positif. Pada level ini, semua pendapat yang ada di dalam dokumen dianggap membahas satu entitas, contohnya seperti sebuah produk.
2. Level pada kalimat.
   Pada level kalimat, akan dlakukan klasifikasi apakah setiap kalimatnya mengungkapkan pendapat yang positif, atau negatif. Jika tidak terdapat pendapat yang diungkapkan, akan dimasukan ke dalam neutral.
3. Entitas dan Level Aspek.
   Pada level aspek, akan dilakukan analisis untuk menentukan sentimen tersebut positif atau negatif berdasarkan aspek yang akan dinilai. Seringkali analisis sentimen pada level dokumen atau kalimat tidak menunjukkan atau menjelaskan apa apa yang menjadikan sentimen tersebut bersifat neagtif atau positif secara lebih rinci.

2.2 Analisis Sentimen Berbasis Aspek

Analisis sentiment berbasis aspek merupakan salah satu bagian dari *opini mining* yang bertujuan untuk mendeteksi polaritas teks tertulis berdasarkan aspek tertentu. Klasifikasi sentimen dalam tingkat dokumen atau kalimat sering kali dianggap tidak cukup diakibatkan klasifikasi yang dilakukan hanya dapat mengidentifikasi sentimen bersifat positif atau bersifat negatif. Untuk analisis lebih lengkap perlu ditemukan aspek dan menentukan sentimen positif atau negatif pada setiap aspek. Dalam melakukan analisis sentimen berbasis aspek, terdapat dua tugas utama, yaitu:

1. Ekstraksi Aspek.
   Pada tahap ini, akan dilakukan ekstaksi aspek yang sebleumnya sudah ditentukan. Contohnya, terdapat kalimat “makanan ini harganya murah.” Aspek dalam kalimat ini adalah “harga” dan entitasnya adalah “makanan.”. Pada entitas “makanan”, tidak menunjukkan aspek umum karena yang akan dievaluasi bukan mengenai makanan secara keseluruhan, tetapi hanya mengenai “harga”.

2. Klasifikasi Aspek Sentimen.
   Pada tahap ini, akan dilakukan penentuan apakah pendapat dari berbagai aspek dapat dimasukkan ke dalam sentiment positif atau negative. Pada contoh kalimat “makanan ini harganya murah.” Sentiment dari aspek “harga” adalah positif[7].

Kategorisasi aspek berfungsi untuk mengetahui aspek yang ada pada setiap ulasannya. Contohnya, di dalam kalimat “walau pelayanan di restoran ini kurang bagus, makanan di restoran ini sangat enak.” Pada kalimat tersebut, kalimat “pelayanan” dan “makanan” diidentifikasi sebagai aspek yang setelahnya

ditentukan sentimen dengan kata “kurang bagus” untuk aspek “pelayanan” yang diidentifikasi mengandung sentimen negatif dan sentimen “sangat enak” untuk aspek “makanan” yang diidentifikasi mengandung sentiment positif. Dalam mengidentifikasi aspek, terdapat beberapa pendekatan, meliputi *frequency based, relation based, supervised learning,* dan *topic modelling.* Untuk penentuan sentiment terhadap aspek dapat dilakukan dengan dua pendekatan, yaitu: *supervised learning* dan *lexicon based* [7].

2.3 *Finance Technology*

Finance technology adalah suatu inovasi pada sector finansial yang mendapat sentuhan teknologi modern. Industri *finance technology* merupakan salah satu layanan jasa keuangan yang mulai popular, khususnya di era serba digital seperti saat ini[8]. Di dalam *finance tecgnology*, terdapat beberapa aktivitas layanan jasa keuangan yang dapat dikelompokan ke dalam lima kategori, yaitu:

1. Manajemen investasi.
   Aktovotas ini mencakup *e-trading* yang memungkinkan penggunanya untuk melakukan investasi secara langsung melalui komputer pada semua jenis asset.
2. Dukungan pasar.
   Aktivitas ini mendukung pengguna untu melakukan proses *e-aggregators*, verivikasi ID *digital* secara lebih sederhana atau lebih efisien.
3. Manajemen risiko,
   Perusahaan *finance technology* yang berpartisipasi di sector asuransi berpotensi mempengaruhi *underwriting*, penetapan harga risiko dan klaim penyelesaian. Manajemen risiko juga memperhatikan komitmen dan registrasi jaminan dan penjaminan dalam operasi kredit.
4. Deposito, pinjaman, dan penambahan modal.

Inovasi *finance technology* yang paling umum dalam kategori ini adalah dalma bidang *crowdfundung* dan *platform* pinjaman P2P (*peer-to-peer)* secara daring.

5. Pembayaran, transfer, kriling, dan penyelesaian.
   Kategori ini meliputi kegiatan yang berkaitan dengan pembayaran secara *mobile* (baik melalui lembaga bank atau non-bank), dompet elektronik (*digital wallet)*, mata uang digital, dan penggunaan teknologi kasbuk atau buku besar yang terdistribusi[8].

Dalam penelitian ini, ulasan yang digunakan adalah ulasan dari aplikasi *finance technology*, khususnya aplikasi yang berfokus pada aktivitas pembayaran dan transfer, seperti beberapa aplikasi dompet digital.

2.4 Preprocessing

Pre-processing adalah bagian terpenting dari setiap *Natural Language Processing* (NLP), karena dalam NLP, setiap karakter, kata, dan kalimat yang akan diindetifikasi pada tahp ini merupakan unit dasar yang akan diteruskan ke semua tahap pemorosesan lebih lanjut, mulai dari komponen analisis dan penandaan seperti penganalisis morfologi dan *part-of-speech taggers* (pemberian tag), melalui aplikasi seperti pengambilan informasi, dan sistem terjemah mesin[9]. Tahap ini merupakan sekumpulan aktvitas dimana dokumen teks akan diproses sebelumnya. Tujuan dari tahap ini agar data yang akan digunakan dalam proses klasifikasi sentimen bersih dari noise. Oleh karena itu, data yang awalnya tidak terstruktur, perlu diubah menjadi data yang terstruktur untuk diproses ke tahap selanjutnya. Pada penelitian ini, akan dilakukan tahap preprocessing, meliputi: *case folding*, *convert emoticon*, *cleansing*, *word normalization*, *convert* negasi, *tokenizing*, *stemming,* dan *stopword removal.*

2.4.1. Case Folding.

Case Folding adalah salah satu proses *preprocessing* yang bertujuan membuat data masukan berbahasa Indonesia yang bercampur antara huruf-huruf alphabet besar dan juga kecil, menjadi sama rata yaitu kecil (*lowercase*)[9]. Berikut adalah contoh inputan berbahsa Indonesia sebelum dilakukan *case folding* dan setelah dilakukan *case folding* pada Tabel 2.1.

Tabel 2.1 Tabel Case Folding

Data *Input* Bahasa Indonesia sebelum dilakukan *case folding*.	Data *Input* Bahasa Indonesia setelah dilakukan *case folding*.
Cara pakenya mudah, terus kalau ada masalah, langsung cepat ditangani. **T**erus memudahkan saya untuk belanjaaa juga :D.	**c**ara pakenya mudah, terus kalau ada masalah, langsung cepat ditangani. **t**erus memudahkan saya untuk belanjaaa juga :D.

Pada Tabel 2.1, kata “Cara” dan “Terus” diubah menjadi huruf kecil pada proses *case folding*.

2.4.2. Convert Emoticon

Emotikon adalah ekspresi wajah yang diwakili dengan berbagai macam kombinasi tanda baca, huruf, dan angka. Emotikon biasanya digunakan untuk mengekspresikan suasana hati yang sedang dirasakan. *Convert emoticon* merupakan salah satu cara untuk mengekspresikan ekspresi yang awalnya menggunakan emoticon, diubah menjadi ekspresi yang digambarkan secara tekstual[10]. Berikut adalah salah satu contoh dari convert emoticon, yaitu : "kesel ya kalau semuanya diabaikan :(", lalu hasil dari convert emoticon tersebut menjadi: "kesel ya kalau semuanya diabaikan sedih". Berikut ini adalah beberapa contoh

emoticon beserta ekspresi yang digambarkan dalam tekstualnya yang dapat dilihat pada Tabel 2.2

Tabel 2.2 Tabel *Convert Emoticon*

Sebelum	Sesudah
😕 :’(:[;(:/): x(--’’ :# :-@ :c :f ;(:v :x :s)’’: *_*	sedih
😊 :] (^_^) ^^v <3 ^^ ^_^ 0 😊 <:} :* (^.^) =) :D :3	senang

Pada Tabel 2.2, emotikon dikategorikan menjadi dua jenis, yaitu emotikon yang dikonversi menjadi senang dan emotikon yang dikonversikan menjadi sedih.

2.4.3. Cleansing

Cleansing adalah proses *preprocessing* yang bertujuan untuk menghapus atau mebersihkan datau ulasan dari simbol-simbol yang tidak dibutuhkan, seperti tanda baca dan angka. Tanda baca akan dihapus, lalu akan diganti dengan spasi. Berikut adalah contoh imputan berbahasa Indoensia sebelum dilakukan *cleansing* dan sesudah dilakukan *cleansing* pada Tabel 2.3

Tabel 2.3 Tabel Cleansing

Data *Input* Bahasa Indonesia sebelum dilakukan *cleansing*.	Data *Input* Bahasa Indonesia setelah dilakukan *cleansing*.
cara pakenya mudah, terus kalau ada masalah, langsung cepat ditangani. terus memudahkan saya untuk belanjaaa juga **:D.**	cara pakenya mudah terus kalau ada masalah langsung cepat ditangani terus memudahkan saya untuk belanjaaa juga

Pada Tabel 2.3, jika dilakukan *cleansing*, terdapat spasi yang akan bertambah, menjadi dua spasi sehingga pada tahap ini juga, penggunaan spasi berlebih akan disaring sehingga hanya terdapat satu spasi. Pada tahap *cleansing*, *emoticon* dan tanda baca akan dihapusnya, oleh karena itu, tanda baca (.) dan (,) dihapuskan.

2.4.4. Word Normalize

Word normalize adalah tahap mengubah kata tidak baku menjadi kata baku. Contoh dari penggunaan kata tidak baku adalah seperti penggunaan huruf vokal berlebihan. Selain itu, kata-kata yang disingkat akan dinormalkan sehingga menbentuk kata baku, seperti kata "sy" berubah menjadi "saya" [10]. Berikut adalah proses *word normalize* yang dapat dilihat pada Tabel 2.4

Tabel 2.4 Tabel Word Normalize

Data *Input* Bahasa Indonesia sebelum dilakukan *word normalize*.	Data *Input* Bahasa Indonesia setelah dilakukan *word normalize*.
cara **pake** mudah terus kalau ada masalah langsung cepat ditangani terus memudahkan saya untuk **belanjaaa** juga	cara **pakai** mudah terus kalau ada masalah langsung cepat ditangani terus memudahkan saya untuk **belanja** juga

Pada Tabel 2.4, kata "pake" dan belanja akan diubah menjadi "pakai" dan "belanja".

2.4.5. Convert Negasi

Convert negasi ialah proses konversi kata-kata negasi yang terdapat pada suatu ulasan. Convert negasi dilakukan dikarenakan kata negasi mempunyai pengaruh dalam mengubah nilai sentimen pada suatu ulasan. Kata negasi yang terdapat pada suatu ulasan akan dihilangkan dan diberikan penanda. Pada tahap ini, akan dilakukan proses mengidentifikasi semua kata negasi. Proses selanjutnya yaitu mengubah polaritas kata-kata yang possinya setelah kata negasi menjadi makna yang sepadan. Contoh kata negasi dala bahasa Indonesia adalah "tidak", "tak", "bukan", dan “tanpa”. [11].

2.4.6. Tokenizing

Tokenizing adalah proses memecah aliran teks menjadi kata, frasa, simbol, atau elemen bermaksa lainnya yang dapat disebut dengan token. Tujuan dari *tokenizing* ini untuk mengeksplorasi kata-kata dalam sebuah kalimat. daftar token akan menjadi masukan untuk diproses lebih lanjut seperti parsing atau penambahan teks[9]. Berikut adalah contoh data inputan sebelum melalui tahap tokenizing dan setelah melalui tahap tokenizing pada Tabel 2.5

Tabel 2.5 Tabel Tokenizing

Data *Input* Bahasa Indonesia sebelum dilakukan *tokenizing*.	Data *Input* Bahasa Indonesia setelah dilakukan *tokenizing*.
cara pakai mudah terus kalau ada masalah langsung cepat ditangani terus memudahkan saya untuk belanja juga	| cara | pakai | mudah | terus | kalau | ada | masalah | langsung | cepat | ditangani | terus | memudahkan | saya untuk | belanja | juga |

Pada table 2., kalimat akan dipisah menjadi token token beradasarkan spasi.

2.4.7. Stemming

Stemming adalah tahap dalam *preprocessing* dimana tahap ini bertujuan untuk mentransformasi kata-kata yang terdapat di dalam suatu dokumen ke kata-kata akarnya (*root word*) dengan menggunakan aturan tertentu. *Stemming* dapat dikatan sebagai proses transformasi kata-kata dalam suatu dokumen menjadi kata dasar. *Stemming*[12], khususnya dalam bahasa Indonesia berbeda dengan *stemming* dalam bahasa Inggris yang hanya menghilangkan sufiks(imbuhan akhiran), *stemming* dalam bahasa Indonesia perlu dilakukan penghilangan prefiks (imbuhan awalan), konfiks(imbuhan awalan dan akhiran yang melekat pada satu kata), dan surfiks(imbuhan akhir), jika ingin mendapatkan kata dasar atau kata akar. Berikut adalah contoh proses dari *stemming* yang ada pada Tabel 2.6

Tabel 2.6 Tabel Stemming

Data *Input* Bahasa Indonesia sebelum dilakukan *stemming.*	Data *Input* Bahasa Indonesia setelah dilakukan *stemming*.
cara pakai mudah terus kalau ada masalah langsung cepat **ditangani** terus **memudahkan** saya untuk belanja juga	| cara | pakai | mudah | terus | kalau | ada | masalah | langsung | cepat | **tangan** | terus | **mudah** | saya untuk | belanja | juga |

Pada Tabel 2.6, kata “memudahkan” dan “ditangai” diproses sehingga berubah menjadi kata dasar “mudah” dan “tangani”.

2.4.8. Stop Removal

Stop removal adalah proses *preprocessing* yang bertujuan untuk menghapus kata-kata yang tidak perlu yang dimana kata-kata tersebut biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna seperti kata-kata “di”, “yang”, “ke”, “oleh” dan lain-lain. Tujuan dari penghapusan kata-kata tersebut adalah untuk mengurangi jumlah kata yang tidak relevan pada teks[9]. Berikut adalah contoh proses dari stop removal yang ada pada Tabel 2.7

Tabel 2.7 Tabel Stopword Removal

Data *Input* Bahasa Indonesia sebelum dilakukan *stopword removal*.	Data *Input* Bahasa Indonesia setelah dilakukan *stopword removal*.
| cara | pakai | mudah | **terus** | **kalau** | ada | masalah | langsung | cepat | tangani | **terus** | mudah | **saya** | untuk | belanja | **juga** |	| cara | pakai | mudah | ada | masalah | langsung | cepat | tangan | memudahkan | untuk | belanja |

Pada Tabel 2.7, kata-kata seperti “terus”, “kalau”, “saya” termasuk ke dalam *stop words* sehingga perlu dihilangkan.

2.5 TF-IDF

Term Weighting TF-IDF (Term Frequency – Invers Document Frequency) adalah salah satu pembobotan yang sering digunakan dan TF-IDF sendiri merupakan gabungan dari *Term Frequency* dan *Inverse Document Frequency*. TF-IDF terdiri dari frekuensi *term* dan *inverse* dokumen yang didapatkan dari membagi seluruh jumlah dokumen terhadap jumlah dokumen yang memiliki *term* tersebut[13].

Untuk dokumen yang jumlahnya tunggal, tiap kalimat dianggap sebagai dokumen. Frekuensi kemunculan kata di dalam dokumen yang diberikan menunjukkan seberapa penting kata tersebut di dalam dokumen itu. Bobot kata semakin besar jika kata tersebut sering muncul dalam suatu dokumen dan bobotnya akan semakin kecil jika kata tersebut muncul dalam banyak dokumen.

Pada perhitungan bobot kata TF-IDF, diguankan rumus untuk menghitung bobot (*w*) masing-masing dokumen terhadap kata kunci dengan rumus sebagai berikut:

$$weight\ (t, d) = tf\ (t, d) \times idf\ (t, D) \qquad (2,1)$$

Dimana definisi dari *t*, *d*, D, *tf (t, d)* , idf (t, D) adalah sebagai berikut:
t = kata,
d = dokumen,
D = kumpulan dokumen atau corpus,
tf (t, d) = frequency t di d, dan
idf (t, D) = invers document frequency t di D

Nilai tf—idf dikatakan tinggi jika salah satu kata *t* muncul berkali-kali dalam jumlah dokumen yang sedikit sedangkan nilai tf-idf dikatakan lebih rendah jika salah satu kata *t* muncul lebih sedikit dalam suatu dokumen atau dalam banyak dokumen. Nilai tf-idf dikatana yang terendah jika kata muncul hampir dalam setiap dokumen. *Term frequency* (tf) menunjukkan seberapa banyak kata yang muncul dalam setiap dokumen. Ini menunjukkan seberapa penting kata tersebut di dalam suatu dokumen. Semakin tinggi bobot tfnya menandakan jika semakin banyak kemunculan kata tersebut dalam dokumen. Rumus tf adalah sebagai berikut:

$$tf = \begin{cases} 1 + log_{10}\,(f_{t,d}), & f_{t,d} > 0 \\ 0, & f_{t,d} = 0 \end{cases} \qquad (2,2)$$

Invers document frequency (idf) menunjukkan jarangnya suatu kata yang muncul. Kata yang jarang muncul berfungsi untuk membedakan satu dokumen dengan dokumen yang lainnya. Rumus idf adalah sebagai berikut:

$$idf = log_{10}\left(\frac{N}{dft}\right) \quad (2,3)$$

N menunjukkan jumlah dokumen dari dokumen, df_t menunjukkan jumlah dari dokumen corpus yang memuat kata *t.* Nilai idf yang tinggi menunjukkan jika kata tersebut jarang muncul, sedangkan nilai idf yang rendah menunjukkan jika kata tersebut sering muncul.

2.6 Norm dan Dot Product

Norm adalah Panjang dari sebuah vector $\boldsymbol{v} = [v_1,v_2,\ldots,v_n\ ]$ pada $\mathbf{R^n}$ atau dapat diddefinisikan sebagai berikut[14]:

$$\|\boldsymbol{v}\| = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2}\ .$$

Jika $\boldsymbol{u} = [u_1, u_2, \ldots., u_n]$ dan $v = v_1,v_2,\ldots,v_n$ adalah vector di $\mathbf{R^n}$ **,** *dot product* dapat direpresentasikan ke dalam matriks berukuran *n* x 1 sebagai berikut:

$$\boldsymbol{u} = \begin{vmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{vmatrix} \text{ dan } \boldsymbol{v} = \begin{vmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{vmatrix}.$$

Dot product dari vector *u* dan *v* dapat direpresentasikan sebagai perkalian matriks *transpose u* dengan matriks *v* sebagai berikut:

$$\boldsymbol{u} \cdot \boldsymbol{v} = \boldsymbol{u}^T \boldsymbol{v} = [u_1 \quad u_2 \quad \cdots \quad u_n] \begin{vmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{vmatrix}$$

$$= [u_1 v_1 \quad u_2 v_2 \quad \cdots \quad u_n v_n].$$

Sifat-sifat *dot product* adalah sebagai berikut:

Jika ***u, v,*** dan ***v*** adalah vector-vektor pada ruang berdimensi dua atau berdimensi tiga dan *k* adalah skalar, sifat-sifatnya akan sebagai berikut:

1. $\boldsymbol{u} \cdot \boldsymbol{v} = \boldsymbol{v} \cdot \boldsymbol{u}$
2. $\boldsymbol{u} \cdot (\boldsymbol{v} + \boldsymbol{w}) = \boldsymbol{u} \cdot \boldsymbol{v} + \boldsymbol{u} \cdot \boldsymbol{w}$
3. $\mathbf{k}\,(\boldsymbol{u} \cdot \boldsymbol{v}) = (\boldsymbol{ku}) \cdot \boldsymbol{v} = \boldsymbol{u} \cdot (\boldsymbol{kv})$.
4. $\boldsymbol{v} \cdot \boldsymbol{v} > 0$ jika $\boldsymbol{v} \neq 0$, dan $\boldsymbol{v} \cdot \boldsymbol{v} = 0$ jika $\boldsymbol{v} = 0$.

2.7 Kernel

Kernel pada *Support Vector Machine* digunakan untuk mengubah data ke ruang dengan dimensi yang lebih tinggi yang disebut dengan ruang kernel [15]. Berikut adalah contoh ilustrasi dari pemisah data dengan menggunakan kernel. Terdapat dua buah data yang dinyatakan sebagai $x_i = (u_1, z_1)$ dan $x_j = (u_2, z_2)$. Diasumsikan fungsi kernel akan dibuat dengan menggunakan kedua data tersebut sebagai berikut:

$$K(x_i, x_j) = (x_i \cdot x_j^{\mathrm{T}})^2$$
$$K(x_i, x_j) = (u_1 u_2 + z_1 z_2)^2$$
$$K(x_i, x_j) = (u_1^2 u_2^2 + z_1^2 z_2^2 + 2\, u_1 u_2 z_1 z_2)$$
$$K(x_i, x_j) = (u_1\sqrt{2}\, u_1 z_1, z_1)\,(u_2\sqrt{2}\, u_2 z_2, z_2)$$
$$K(x_i, x_j) = \emptyset(x_i)\, \emptyset(x_j)^T$$

Nilai K yang sudah disebutkan di atas telah mendefinisikan pemetaan ke ruang dengan dimensi yang lebih tinggi sebagai berikut:

$$\emptyset(x_i) = \{u_1, \sqrt{2}u_1 z_1, z_1\} \qquad (2,4)$$

Contoh numerik dari kernel adalah sebagai berikut. Misal xi = (5, 3) dan x_j = (2, 5), sehingga:

$K(x_i, x_j) = (x_i \cdot x_j^T)^2$

$= (5 \cdot 2 + 3 \cdot 5)^2$

$= (10 + 15)^2$

$= 25^2$

$K(x_i, x_j) = 625$

Berikut adalah fungsi-fungsi kernel yang bisa diterapkan dalam metode *Support Vector Machine*:

1. Kernel Linear.

   Kernel linear adalah fungsi kernel yang paling sederhana. Kernel ini cocok digunakan ketika terdapat banya fitur dikarenakan pemetaan ke ruang dimensi yang lebih tinggi tidak benar-benar meningkatkan kinerja, contohnya pada klasifikasi teks. Dalam klasifikasi teks, baik jumlah dokumen maupun jumlah fitur (kata) sama-sama besaar. Berrikut adalag persamaan dari kernel linear:

$$K(x_i, x_j) = (x_i \cdot x_j^T) \quad (2,5)$$

2. Kernel *Radial Basic Function*

   Kernel *Radial Basic Function* adalah fungsi yang digunakan saat data tidak dapat dipisahkan secara linear. RBF kernel memiliki dua parameter; Gamma dan Cost. Tugas dari parameter Cost atau C sebagai pengoptimalan *Support Vector Machine* untuk menghindari terjadinya klasifikasi yang salah di setiap sampel dalam dataset untuk data latih. Tugas parameter Gamma sebagai penentu pengaruh dari satu sampel dataset untuk data latih pada garis pemisahnya. Berikut adalah persamaan dari kernel RBF:

$$K(x_i, x_j) = \exp[-\gamma \|x - z\|^2] \quad (2,6)$$

3. Kernel *Polynomial*

Kernel *polynomial* adalah fungsi kernel yang digunakan saat data tidak dapat dipisahkan secara linear. Kernel ini cocok digunak untuk permasalah dimana semua data training dataset dinormalisasi. Berikut adalah persamaan dari kernel *polynomial*:

$$K(x_{i,}\, x_j) = (x_i \cdot x_j^T)^d \qquad (2,7)$$

2.8 Grid Search Cross Validation

Grid Search Cross Validation (Grid Search CV) ialah salah satu proses untuk melakukan pemilihan *hyperpaarmeter* terbaik suatu model yang diberikan. *Hyperparameter* sendiri merupakan parameter yang ditentukan tanpa proses uji, dengan kata lain, paramater yang tdak ditentukan oleh mesin. *Hyperparameter* yang diperolah merupakan parameter terbaik yang akan dimasukkan pada model. Pada *Grid Search CV*, akan dilakukan kombinasi dari hyperparameteri yang telah ditentukan dan menghitung rata-rata nilai *cross validation* dari setiap kombinasinya. Kombinasi dengan rata-rata nilai *cross validation* akan dimasukkan ke dalam model.

Contoh sederhana untuk menggambarkan cara kerja *Grid Search CV* ini adalah, misal *hyperparameters* X = [3,4] dan Y= [1,0], kemudian *Grid Search CV* akan melakukan kombinasi X dan Y dengan hasil [3,1], [3,0], [4,1], dan [4,0]. Setelah itu, lalu akan dilakukan pemilihan kombinasi terbaik berdasarkan rata-rata nilai *cross validation* tertinggi.

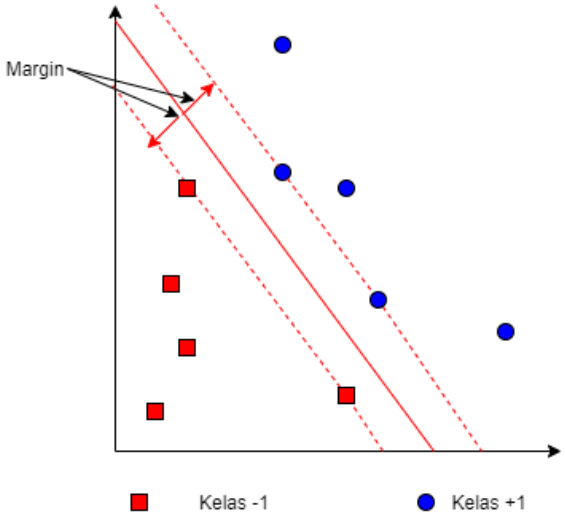
2.9 *Support Vector Machine*

Support Vector Machine atau disingkat menjadi SVM adalah salah satu metode klasifikasi yang diperkenalkan oleh Vladimir Vapnik, Boser, dan Guyon pada tahun 1992. Sejak dikenalkan pada tahun 1992, SVM telah menemukan jalan

ke dalam berbagai macam aplikasi, seperti prediksi cuaca, prediksi perkiraan daya, klasifikasi cacar, pengenalan pembicara, identifikasi tulisan tangan, pemrosesan gambar dan audio, analisis video, dan diagnosis medis[16]. SVM sendiri juga banyak digunakan dalam pemrosesan bahasa alamiah, seperti salah satunya dalam melakukan analisis sentiment. SVM merupakan salah satu metode *supervised*, dimana *supervised* adalah pendekatan yang mana sudah terdapat data yang sebelumnya data tersebut sudah dilatih, dan terdapat variable yang ditargetkan sehingga tujuan dari pendekatan ini adalah mengelompokan suatu data ke data yang sudah ada. Oleh karena itu, data yang sudah beri label sangat dibutuhkan.

Metode SVM akan mengubah data dari ruang yang berdimensi lebih tinggi dan akan menemukan *hyperplane* yang terbaik. *Hyperplan* sendiri merupan bidang datar penentu yang akan memisahkan dua buah kelas di dimensi *n*[16]. SVM juga akan mengembalikan parameter *hyperplane* yang optimal yang sama. Di dalam metode SVM, perlu dilakukan pencarian *hyperplane* terbaik, untuk mencarinya, dapat dilakukan dengan cara mengukur *margin hyperplane* tersebut. Margin sendiri merupakan jarak antar *hyperplane* dengan *pattern* atau motif terdekat dari masing-masing kelas. *Pattern* atau motif terdekat dengan *hyperplane* dinamakan *support vector*. *Support vector* ditemukan setelah langkah pengoptimalan yang melibatkan fungsi tujuan yang diatur oleh oleh istilah *error term* dan Batasan menggunakan Langrangian *relaxation*.

Misalkan data latih akan dinyatakan sebagai (x_i , y_i) dimana I = 1, 2, 3, …, n. x_i = [x_{i1} , $x_{i2,\ldots,}$ xi_j] merupakan vector baris dari fitur ke-i di ruang dimensi ke-j dan y_i adalah label dari x_i yang didefinisikan sebagai $y_i \in \{+1, -1\}$. Diasumsikan kedua kelas -1 dan +1 dapat dipisahkan secara linear oleh *hyperplane*. Pada Gambar 2.1 *hyperplane* ditunjukkan dengan garis lurus berwarna merah. Data yang berada di atas *hyperplane* adalah kelas +1 dan data yang berada di bawah *hyperplane* adalah kelas -1.



Gambar 2.1 Contoh hyperplane Dua Dimensi.

Persamaan *hyperplane* didefinisikan sebagai berikut:

$$f(x) = \boldsymbol{w} \,.\, \boldsymbol{x} + b, \qquad (2,10)$$

dimana:

$\boldsymbol{w}$ = parameter bobot,

$\boldsymbol{x}$ = vector input,

b = bias.

Vektor $\boldsymbol{w}$ memiliki arah tegak lurus dengan *hyperplane*. Jika nilai b *berubah*, *hyperplane* akan berubah juga. *Hyperplane* terbaik adalah *hyperplane* yang terletak di tengah-tengah antara dua set objek dari du akelas. Oleh karena itu, diperlukan pencarian *hyperplane* terbaik dengan mendapatkan nilai *margin* terbesar. *M*argin terbesar dapat ditemukan dengan memaksimalkan nilai jarak antar *hyperplane* dari titik terdekatnya. *Pattern* yang memenuhi kelas -1 adalah *pattern* yang memenuhi persamaan $\boldsymbol{w} \,.\, \boldsymbol{x_i} + b = -1$ dan *pattern* yang memenuhi kelas +1 adalah *pattern* yang memenuhi persamaan $\boldsymbol{w} \,.\, \boldsymbol{x_i} + b = 1$.

Support vector direpresentasikan sebagai titik (x, y). *Hyperplane* sebagai berikut:

$$Ax + By + C = 0, \qquad (2,11)$$

Dengan rumus jarak sebagai berikut:

$$d = \frac{|Ax+By+C|}{\sqrt{A^2+B^2}}$$

Persamanaan 2,11 diubah menjadi bentu *dot product* pada vector sehingga menjadi:

$$[A\ B]\begin{bmatrix} x \\ y \end{bmatrix} + C = 0.$$

Misalkan $\boldsymbol{w} = [A\ B]$ dan $\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix}$ dan b = C, maka diperoleh:

$$d = \frac{|Ax+By+C|}{\sqrt{A^2+B^2}} = \frac{|\boldsymbol{w}.\ \boldsymbol{x}+b|}{\sqrt{\boldsymbol{w}^2+C^2}} = \frac{|\boldsymbol{w}.\ \boldsymbol{x}+b|}{\sqrt{\boldsymbol{w}^2}} = \frac{|\boldsymbol{w}.\ \boldsymbol{x}+b|}{||\mathbf{w}||}$$

Nilai margin dapat dicari menggunakan nilai tengah antara jarak kedua kelas sebagai berikut:

$$\textit{margin} = \frac{1}{2}(d^+ - d^-)$$

$$= \frac{1}{2}\left(\frac{|\boldsymbol{w}\ .\boldsymbol{x}\mathbf{1}+b|}{||\mathbf{w}||} - \frac{|\boldsymbol{w}\ .\boldsymbol{x}\mathbf{2}+b|}{||\mathbf{w}||}\right)$$

$$= \frac{1}{2}\left(\frac{\mathbf{1}}{||\mathbf{w}||} - \frac{(-\mathbf{1})}{||\mathbf{w}||}\right)$$

$$= \frac{\mathbf{1}}{||\mathbf{w}||}\ , ||\boldsymbol{w}|| \neq 0,$$

dimana:

d^+: jarak antara *hyperplane* terhadap kelas +1,

d^-: jarak antara *hyperplane* terhadap kelas -1.

Setiap kelas harus ditambahkan Batasan pada data dari masing-masing kelas supaya tidak masuk ke dalam *margin*, batasannya sebagai berikut:

$\boldsymbol{w} \cdot \boldsymbol{x_i} + b \leq -1$, jika $y = -1$

$\boldsymbol{w} \cdot \boldsymbol{x_i} + b \geq +1$, jika $y = +1$,

atau dapat ditulis sebagai berikut:

$$y_i\ (\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1 \geq 0, \forall 1 \leq i \leq n, i \in N.$$

Memaksimalkan nilai margin ekuivalen dengan meminimumkan $\|\boldsymbol{w}\|^2$**.** Pencarian *hyperplane* terbaik dengan nilai *margin* terbesar dapat dirumuskan menjadi masalah optimasi pemograman kuadratik sebagai berikut:

$$\text{max margin} = \min \frac{1}{2} \|\boldsymbol{w}\|^2\text{,}$$

dengan kendala:

$$y_i\ (\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1 \geq 0, \forall 1 \leq i \leq n, i \in N.$$

Masalah ini dapat diselesaikan dengan mengubah persamaan ke dalam fungsi *largrage*:

$$\min L_p\ (\boldsymbol{w}, b, \alpha) = \frac{1}{2} \|\boldsymbol{w}\|^2 - \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{\alpha}_I\ [y_i\ (\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1],$$

dimana:

L_p : fungsi *lagrange*(*primal problem),*

α_i : nilai dari koefisien *lagrange,* $\alpha_i \geq 0$ dengan i = 1, 2, 3, …, n.

Fungsi L_p diminumkan terhadap $\boldsymbol{w}$ dan b dimaksimaklan terhadap α, sehingga akan dicari turunan pertama dari fungsi L_p terhadap $\boldsymbol{w}$ dan b, akan didapat:

1. Turunan pertama fungsi L_p terhadap $\boldsymbol{w}$

$$\frac{\partial}{\partial \boldsymbol{w}} L_p\ (\boldsymbol{w}, b, \alpha) = 0.$$

Akan didapatkan:

$$\min L_p\ (\boldsymbol{w}, b, \alpha) = \frac{1}{2} \|\boldsymbol{w}\|^2 - \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{\alpha}_I\ [y_i\ (\boldsymbol{w} \cdot \boldsymbol{x_i} + b)] + \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{\alpha},$$

$$\frac{\partial}{\partial \boldsymbol{w}} L_p\ (\boldsymbol{w}, b, \alpha) = \boldsymbol{w} - \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{\alpha}_i y_i \boldsymbol{x}_i$$

$$\Leftrightarrow \qquad 0 = \boldsymbol{w} - \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{\alpha}_i y_i \boldsymbol{x}_i$$

$$\Leftrightarrow \qquad \boldsymbol{w} = \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{\alpha}_i y_i \boldsymbol{x}_i \qquad (2{,}12)$$

2. Turunan pertama fungsi L_p terhadap b

$$\frac{\partial}{\partial \boldsymbol{w}} L_p\,(\boldsymbol{w}, b, \alpha) = 0.$$

Akan didapatkan:

$$\min L_p\,(\boldsymbol{w}, b, \alpha) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{\alpha}_I\,[y_i\,(\boldsymbol{w}\cdot \boldsymbol{x_i} + b)] + \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{\alpha},$$

$$\frac{\partial}{\partial \boldsymbol{w}} L_p\,(\boldsymbol{w}, b, \alpha) = \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{\alpha}_i y_i \boldsymbol{x}_i$$

$$\Leftrightarrow \qquad 0 = \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{\alpha}_i y_i. \qquad (2{,}13)$$

Rumus lagrange L_p (*primal problem*) diubah menjadi L_D (*dual problem*).

$$\textit{Maks}\ L_D(\alpha) = \frac{1}{2}\left(\sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i\right)\left(\sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i\right) - \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{\alpha}_i y_i \left(\left(\sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i\right)\boldsymbol{x_i} + b\right) + \alpha_i$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i y_i\, \alpha_j y_j\, (x_i \cdot x_j) - \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i y_i\, \alpha_j y_j\, (x_i \cdot x_j) - b$$

$$= \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i y_i\, \alpha_j y_j\, (x_i \cdot x_j), \qquad (2{,}14)$$

dengan kendala,

$\sum_{i=1}^{n} \alpha_i y_i = 0,\ \alpha_i \geq 0.$

Nilai α_i didapatkan dari hasil perhitungan subtitusi kendala pada persamaan (2,14). Nilai α_i akan digunakan untuk mencari nilai $\boldsymbol{w}$. Setiap titik data selalu terjadi $\alpha_i = 0$. Titik-titik data dimana $\alpha_i = 0$ tidak akan muncul dalam perhitungan mencari

nilai $\boldsymbol{w}$ sehingga tidak berperan dalam memprediksi data baru. Data lain dimana $\alpha_i > 0$ disebut dengan *support vector*.

Dilakukan *sign*{$f(x)$} untuk menguji data baru menggunakan model yang sudah dilatih. Substitusikan persamaan (2,12) ke persamaan (2,10) dan menggunakan kernel linear

$K(\boldsymbol{x_i}, \boldsymbol{x_j}) = \boldsymbol{x} \cdot \boldsymbol{x}^{\boldsymbol{T}}$ sehingga akan didapatkan:

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i (\boldsymbol{x}_{\mathrm{i}}^{T} \cdot \boldsymbol{x}) + b. \qquad (2,15)$$

Lalu dilakukan distribusi persamaan (2,15) ke dalam y_i $f(x)$ = 1 sehingga akan didapatkan:

$$y_i \sum_{m \in S} \alpha_m y_m \boldsymbol{x}_{\mathrm{m}}^{T} \cdot \boldsymbol{x_i} + b = 1,$$

dimana S adalah himpunan indeks *support vector*. Nilai b diperoleh sebagai berikut:

$$y_i \left(\sum_{i=m}^{S} \alpha_m y_m \boldsymbol{x}_{\mathrm{m}}^{T} \cdot \boldsymbol{x_i} + b \right) = 1$$

$$\Leftrightarrow \quad y_i y_i \left(\sum_{i=m}^{S} \alpha_m y_m \boldsymbol{x}_{\mathrm{m}}^{T} \cdot \boldsymbol{x_i} + b \right) = y_i$$

$$\Leftrightarrow \quad \left(\sum_{i=m}^{S} \alpha_m y_m \boldsymbol{x}_{\mathrm{m}}^{T} \cdot \boldsymbol{x_i} + b \right) = y_i$$

$$\Leftrightarrow \quad b = y_i - \sum_{i=m}^{S} \alpha_m y_m \boldsymbol{x}_{\mathrm{m}}^{T} \cdot \boldsymbol{x_i}$$

$$\Leftrightarrow \quad b = \frac{1}{N_s} \sum_{i \in S} \left(\sum_{i=m}^{S} \alpha_m y_m \boldsymbol{x}_{\mathrm{m}}^{T} \cdot \boldsymbol{x_i} \right)$$

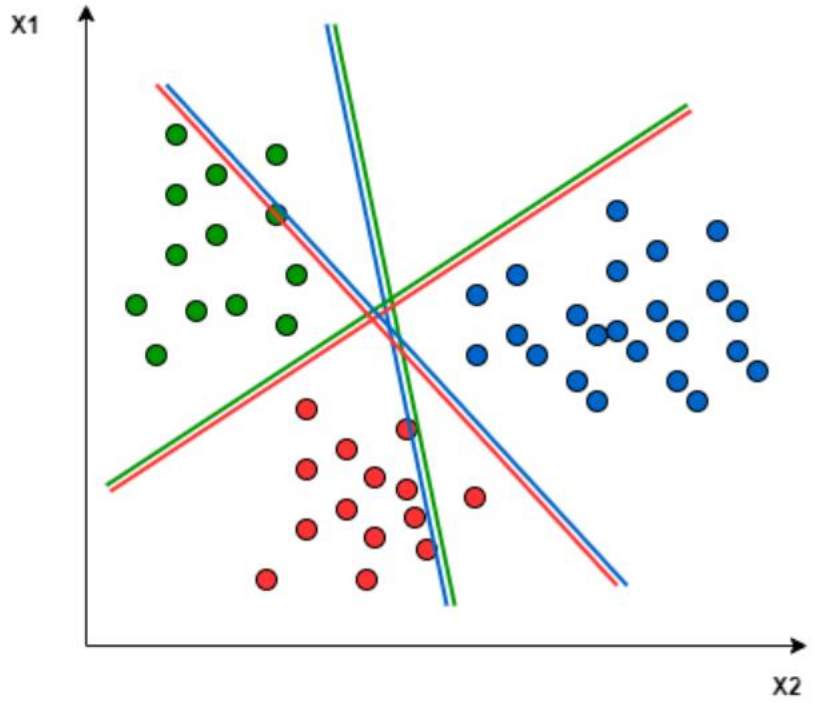
(2,16)

Dimana N_S adalah jumlah *support vector*.

Pada dasarnya, SVM hanya dapat mengklasifikasikan data dalam dua kelas, tidak dapat mendukung klasifikasi lebih dari du akelas atau *multiclass*. Namun, untuk klasifikasi *multiclass* dapat dilakukan dengan prinsip yang sama dengan

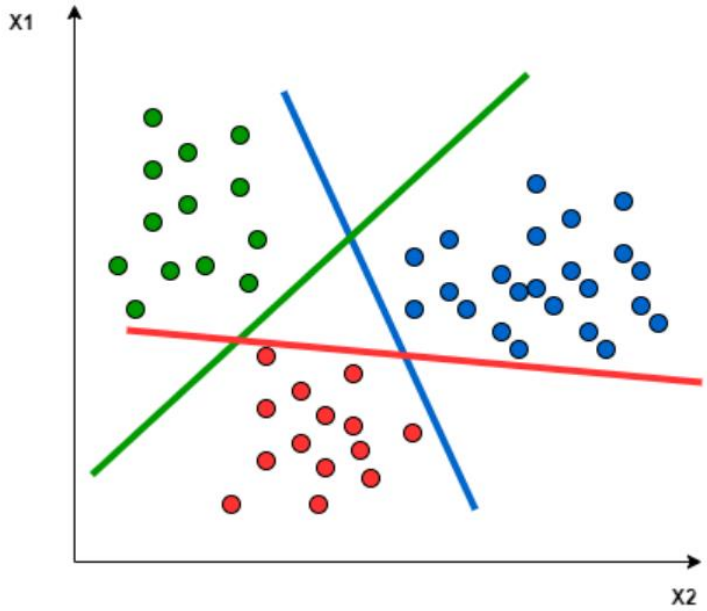
memecah masalah *multiclass* menjadi beberapa masalah pada klasifikasi biner. Dalam klasifikasi *multiclass*, dibagi menjadi dua pendekatan, yaitu:

1. One vs One: Pendekatan One vs One akan memecah masalah *multiclass* menjadi beberapa klasifikasi biner dan pengklasifikasian biner dilakukan per setiap pasangan kelas sehingga *hyperplane* yang akan terbentuk hanya memisahkan dua kelas dengan mengabaikan kelas ketiga. Pendekatan One vs One dapat dilihat pada Gambar 2.2 Pendekatan One vs One



Gambar 2.2 Pendekatan One vs One

2. One vs Rest: Pendekatan One vs Rest dilakukan dengan perincian yang diatur ke pengklasifikasian biner per setiap kelas. Pada pendekatan ini, dibutuhkan *hyperplane* yang akan memisahkan antara kelas dengan kelas yang lainnya secara sekaligus. Pendekatan One vs Rest dapat dilihat pada Gambar 2.3 Pendekatan One vs Rest



Gambar 2.1 Pendekatan One vs Rest

Salah satu masalah yang terjadi di dunia nyata untuk pengklasifikasian *multiclass* menggunakan SVM adalah klasifikasi teks [17], seperti pengkasifikasian aspek pada analisis sentimen berbasis aspek.

2.10 Evaluasi Model

Evaluasi model perlu dilakukan untuk mengetahui seberapa baik suatu model dapat mengklasifikasikan suatu kelas. Dalam penelitian ini, metode yang dipilih ialah dengan menggunakan *confussion matrix*. *Confussion Matrix* merupakan sebuah tabel yang menyatakan banyaknya data uji yang benar dan salah diklasifikasikan[13]. Parameter yang digunakan pada data uji yaitu TP (*True Positive*), FN(*False Negative*), TN(*True Negative*), dan FP(*False Negative*) yan dapat dilihat pada Tabel 2.8

Tabel 2.8 Parameter Confussion Matrix

Aktual	Prediksi	
	Negatif	Positif
Negatif	TN	FN
Positif	FP	TP

Dengan menggunakan *confussion matrix*, dapat menghasilkan nilai akurasi. Nilai akurasi digunakan untuk mengukur seberapa akurat suatu model dalam melakukan pengklasifikasian suatu kelas dengan benar. Rumus untuk menghitung nilai akurasi adalah:

$$\text{Akurasi} = \frac{TN+TP}{TN+FP+TP+FN} \qquad (2,17)$$

Contoh kasus sederhana dalam penerapan *confussioon matrix* adalah misalkan, di dalam Dinas Lingkungan Hidup dan Kebersihan ingin mengukur kinerja dari sebuah mesin pemisah yang bertugas untuk memisahkan sampah organic dari semua sampah yang ada. Untuk mengujinya, petugas akan meamsukkan 100 sampah organic dan 900 sampah lainnya. Hasil mesin tersebut memisahkan 150 yang terdeteksi sebagai sampah organik. Kemudian, setelah diperiksa kembali oleh petugas, dari 150 sampah tersebut, hanya terdapat 100 barang yang merupakan sampah organic, sedangkan 50 lainnya adalah sampah lain. Dengan begitu, perhitungan dari akurasi adalah sebagai berikut:

$$\text{Akurasi} = \frac{100+900}{100+50+50+900} = \frac{1000}{1100} = 0,90$$

Contoh kasus sebelumnya dapat diterapkan jika kelas yang digunakan sebanyak dua kelas. Jika dalam proses klasifikasi terdapat lebih dari dua kelas, bentuk dari *confusion matrix* yang terbentuk akan menjadi seperti pada Tabel 2.9

Tabel 2.9 Tabel Confusion Matrix Multi Class

Aktual	Prediksi		
	A	B	C
A	**6**	0	1
B	3	**9**	1
C	1	0	**10**

Dengan *Confusion Matrix*, selain dapat menghitung nilai akurasi, matriks tersebut dapat digunakan dalam menghitung nilai Presisi dan Recall. Presisi adalah pecahan dari *True Positive* dibagi dengan jumlah data yang diprediksi secara positif (kolom jumlah dari prediksi positif). Untuk menghitung Presisi, dapat digunakan dengan rumus:

$$Presisi = \frac{TP}{TP+FP} \quad (2,18)$$

Recall adalah pecahan elemen *True Positive* dibagi dengan jumlah data yang diklasifikasikan secara positif (jumlah baris positif yang sebenarnya) [18]. Untuk menghitung Recall, dapat digunakan dengan rumus:

$$Recall = \frac{TP}{TP+FN} \quad (2,19)$$

Hasil pengujian penelitian sering disajikan dengan *confusion matrix*. *Confusion matrix* dengan *k* kelas ialah tabel kontigensi k x k dengan sel-sel [i,j] dimana (i=1,… k, j=1,…,k). *Confusion matrix* dengan k x k dapat direpresentasikan

sebagai himpunan k *confusion matrix* biner satu setiap kelas Ci. Contohnya dapat dilihat pada Gambar 2.4

Aktual	Prediksi		
	A	B	C
A	**6**	0	1
B	3	**9**	1
C	1	0	**10**

Aktual	Prediksi	
	A	Bukan A
A	6	(0 + 1) = 1
Bukan A	(3 + 1) = 4	(9 + 1 + 0 + 10) = 20

Gambar 2.2 Bentuk Confusion Matrix Biner pada Multiclass

Dalam praktiknya, transformasi *confusion matrix* ke dalam bentuk binernya tidak diperlukan, tetapi transformasi ini dapat membantu menjelaskan bagaimana indikator kinerja klasifikasi yang berbeda dihitung dari *confusion matrix* untuk *multiclass*. [19]

2.11 *Web Scrapping*

Web scrapping merupakan proses pengambilan sebuah dokumen semi-terstruktur yang diperoleh dari internet, biasanya berupa halaman-halaman web dalam bahasa markup seperti HTML (IHypertext Markup Language) dan XHTM (eXtensible Hypertext Markup Language) dan menganalisis dokumen tersebut untuk diambil data-data tertentu yang akan digunakan pada kepentingan yang lainnya[13]. Manfaat dari *web scrapping* adalah supaya informasi yang akan dikeruk lebih terfokus sehingga memudahkan dalam pencarian sesuatu. Dalam penelitian ini, *tools* yang digunakan dalam pengambilan data menggunakan cara *web scrapping* adalah dengan menggunakan ekstenti *web scrapper*. Berikut adalah langka-langkah dalam *web scraping*:

1 Akses halaman web yang hendak dikeruk informasinya.
2 Pilih elemen data yang ada di dalam halaman web tersebut.

3 Tools yang digunakan pun akan memulai melakukan ekstraksi dan transformasi bila diperlukan.
4 Setelah data berhasil diekstraksi, simpang data tersebut menjadi dataset terstruktur.

2.12 Python

Python adalah bahasa pemograman tingkat tinggi yang banyak digunakan untuk pemograman tujuan umum. *Python* dibuat oleh Guido van Rossum dan pertama kali dirilis pada tahun 1991[20]. *Python* memiliki fitur sistem dinamis dan dapat memanajemen memori dengan otomatis. Selain itu, pytjon juga mendukung beberapa paradigma pemograman; termasuk pemograman berorientasi objek, imperatif, fungsional, dan gaya prosedural. Salah satu kelebihan dari bahasa pemrograman *Python* adalah memilik *library* yang luas dan banyak. Bahasa pemrograman *Python* memiliki *library* yang luas dengan beragam modul yang siap untuk digunakan. Di dalam *library* tersebut, terdapat berbagai macam kode untuk berbagai macam keperluan, seperti *regular expression*, *documentation-generation*, *unit-testing*, *databases*, CGI atau *Common Gateway Interface*, dan email. Dengan kelebihan ini, penggunanya tidak perlu menulis kole lengkap secara manual.

2.13 Pemograman Terstruktur

Pemograman terstruktur atau dapat disebut dengan analisis terstruktur ialah analisis yang memperlakukan data dan proses yang melakukan transformasi data tersebut sebagai entitas yang terpisah[21]. Objek-objek data dimodelkan dengan cara mengdefinisikan atribut-atribut, serta relasi-relasinya. Proses-proses tersebut akan memanipulasi objek-objek data yang sudah dimodelkan sedemikian rupa sehingga dapat menunjukkan bagaimana caranya mereka mentransformasi data saat objek-objek data mengalir di dalam suatu sistem yang akan dibangun. Di dalam pemodelan analisis terstruktur, terbagi menjadi:

1. Diagram Konteks

Diagram Konteks merupakan kejadian tersendiri dari suatu diagram aliran data. Diagram konteks akan mebentuk satu lingkaran, dimana satu lingkaran tersebut akan merepresentasikan seluruh sistem. Diagram Konteks ini berupa suatu pandangan yang mencakup masukan-masukan dasar, sistem-sistem, dan keluarannya. Diagram konteks juga merupakan tingkatan tertinggi dalam suatu diagram aliran data dan hanya memuat satu proses yang menunjukkan sistem secara keseluruhan. Diagram konteks pun tidak memuat penyimpanan data dan terlihat sederhana untuk dibuat. Entitas-entitas eksternal serta aliran data-data dari sistem merupakan hasi analasis dari wawancara dengan pengguna dan sebagai analisis dokumen.

2. DFD (*Data Flow Diagram*)

DFD adalah peta aliran informasi untuk setiap proses atau setiap sistem. DFD juga dapat diartikan sebagai gambaran arus informasi yang akan diproses dan akan diinput menuju sebuah keluaran tertentu. DFD akan berfokus pada arus informasi, asal dan tujuan data, hingga bagaimana data tersebut akan disimpan. DFD biasanya akan digambarkan dalam bentuk hierarki atau bertingkat. Diagram Konteks merupakan DFD tingkat 0, dimana akan menggambarkan sistem secara keseluruhan. DFD tingkat selanjutnya adalah penghalusan dari diagram konteks, dimana akan memberikan gambaran yang semakin merinci dari diagram konteks, dan hal tersebut akan terus berlanjut ke tingkat-tingkat selanjutnya.

Berikut adalah beberapa komponen yang digunakan dalam DFD:

a. Entitas luaar.
   Entitas luar akan digambarkan dengan persegi panjang. Persegi Panjang tersebut disebut dengan terminator. Terminator akan mewakili entitas luar yang berkomunikasi dengan sistem yang sedang dikembangkan.

Terminator dibagi menjadi dua: terminator sumber dan terminator tujuan.

b. Proses
Komponen proses akan menggambarkan bagian dari sistem yang akan ditransformasi atau diubah dari inputan atau masukan menjadi output atau keluaran. Komponen ini biasanya digambarkan dengan bentuk bulat.

c. Penyimpanan data.
Komponen dari penyimpanan data akan digunakan untuk membuat model dari sekumpulan paket data dan diberi nama dengan kata benda jamak. Simpanan data biasanya akan digambarkan dengan dua garis sejajar.

d. Arus data.
Suatu arus data akan digambarkan dengan anak panah yang menunjukkan arah menuju ke dan keluar dari suatu proses. Alur data berfungsi untuk menerangkan atau menjelaskan perpindahan data atau paket data dari suatu bagian sistem ke bagian sistem yang lainnya[22].

3. Kamus Data

Kamus data merupakan katalog fakta mengenai data dan kebutuhan -kebutuhan informasi dari suatu sistem. Kamus data dapat digunakan sebagai alat komunikasi antara analisis sistem dengan pemakai sistem mengenai data yang mengalir di sebuah sistem, yaitu mengenai data yang masuk ke dalam suatu sistem dan tentang informasi yang dibutuhkan oleh pengguna sistem. Kamus data berisi mengenai beberapa komponen, seperti:

a. Nama : nama data.
b. Digunakan pada : merujuk ke proses-proses yang berkaitan dengan data.

c. Deskripsi : deskripsi dari data.
d. Informasi tambahan, seperti tipe data, nilai data, batas nilai data, dan komponen lainnya yang membentuk data.

4. *Flow Chart*.

Flowchart atau bagan alir merupakan skema atau bagan yang menunjukkan alir atau *flow* di dalam suatu program secara logika. *Flowchart* banyak digunakan dalam presentasi untuk menjelaskan secara visual urutan langkah agar penggunanya lebih mudah memahami. *Flowchart* biasanya memudahkan dalam penyelesaian suatu masalah, khususnya masalah yang perlu dipahami dan dievaluasi lebih lanjut. *Flowchart* digambarkan menggunakan anotasi bidang-bidang geometri, seperti lingkaran, persegi empat, belah ketupat, oval, dan sebagainya dan digunakan untuk merepresentasikan langkah-langkah kegiatan beserta urutannya dengan menghubungkan masing-masing langkah menggunakan tanda panah[]. Dalam jenisnya, *flowchart* dapat dikelompokkan dalam beberapa jenis berdasarkan fungsi dan prosesnya. *Flowchart* terbagi menjadi lima jenis, yaitu:

1. *Flowchart* Sistem.

   Flowchart sistem adalah bagan yang menunjukkan alur kerja atau apa yang akan dilakukan atau dikerjakan di dalam sistem secara keseluruhan dan akan menjelaskan urutan dari langkah-langkah yang ada di dalam sistem. Dengan kata lain, *flowchart* sistem merupakan desskripsi secara grafik dari urutan langkah-langkah yang tergabungkan hingga membentuk suatu sistem. *Flowchart* sistem terdari atas data yang mengalir melalui sistem dan proses yang mentrasformasikan data tersebut.

2. *Flowchart* Dokumen.

Flowchart dokumen atau disebut *Paperwork* akan menelusuri alur data yang ditulis melalui sistem. Fungsi utama dari *flowchart* dokumen adalah untuk menelusuri alur *form* dan laporan sistem dari satu bagian ke bagian yang lainnya, baik alur *form* dan laporan diproses, dicatat, lalu disimpan.

3. *Flowchart* Skematik

Flowchart skematik mirip dengan *flowchart* sistem yang menggambarkan suatu sistem atau prosedur. *Flowchart* skematik juga tidak hanya menggunakan symbol-simbol *flowchart* standar, tetapi *flowchart* skematik juga menggunakan gambar-gambar komputer, *peripheral*, form-fom atau peralatan lain yang digunakan di dalam sistem. *Flowchart* ini digunakan sebagai alat komunikasi antara analisis sistem dengan orang yang asing dengan simbol-simbol *flowchart* yang konvensional. Pemakaian gambar ini digunakan segabai ganti dari simbol-simbol *flowchart* yang dapat menghemat waktu orang yang asing dengan simbol *flowchart* tersebut sebagai simbol abstrak sebelum mengerti *flowchart*. Penggunaan gambar ini dapat mengurangi kemungkinan salah persepsi atau salah arti mengenai suatu sistem.

4. *Flowchart* Program.

Flowchart program merupakan hasil dari *flowchart* sistem. *Flowchart* program juga merupakan keterangan yang lebih rinci mengenai bagaimana setiap langkah atau prosedur sesungguhnya akan dilaksanakan. *Flowchart* program akan menunjukkan setiap langkag program atau langkah-langkah dalam urutan yang tepat saat terjadi. Dalam analisis sistem, biasanya menggunakan *flowchart* program untuk menggambarkan urutan tugas-tugas pekerjaan dalam suatu prosedu atau operasi.

5. *Flowchart* Proses.

Flowchart proses merupakan Teknik penggambaran rekaya industrial yang akan membagi dan menganalisis langkah-langkah selanjutnya dalam suatu sistem atau dalam suatu prosedur. *Flowchart* proses digunakan untuk mempelajari atau memahami dan mengembangkan proses-proses kerja utama untuk industry. Di dalam analisis sistem, *flowchart* akan digunakan secara efektif untuk menelusuri alur suatu laporan atau *form*[23].