BAB II

LANDASAN TEORI

2.1 Transaksi *E-Commerce*

E-commerce atau perdagangan *online* dapat diartikan sebagai proses menjual dan membeli produk-produk secara elektronik oleh konsumen dan dari perusahaan ke perusahaan dengan komputer sebagai perantara transaksi bisnis (Loudon & Traver, 2007). *E-Commerce* lebih mengacu pada teknologi seperti *mobile commerce*, transfer dana elektronik, manajemen rantai pasokan, pemasaran internet, pemrosesan transaksi secara *online*, *electronic data interchange* (EDI), sistem manajemen persediaan, dan sistem pengumpulan data otomatis (Shahriari, Shariari, & Geiji, 2015).

Sedangkan transaksi diartikan sebagai persetujuan jual beli dari perspektif perdagangan (KBBI Kemendikbud, 2008). Transaksi *e-commerce* (Setiawati, 2015) merupakan transaksi yang dilakukan penjual dan pembeli secara *online* melalui media internet, tidak ada perjumpaan langsung antara pembeli dan penjual.

Tweet transaksi *online* mempunyai ciri berupa *tweet* yang berhubungan dengan informasi aktifitas sebelum pembelian (berminat, memesan, dan membatalkan), aktifitas transaksi pembelian dan aktifitas pengiriman barang (Kodra & Purwarianti, 2013). Sedangkan *tweet* iklan dicirikan dengan *tweet* yang mengandung unsur AIDCA (Muis & Affandes, 2015) AIDCA sendiri merupakan akronim dari *attention* (perhatian), *interest* (minat), *desire* (kebutuhan), *conviction* (keinginan) dan *action* (tindakan). Teori ini merupakan kriteria dari sebuah iklan yang baik. Selain itu, iklan pada Twitter juga merupakan *tweet* yang mengandung nama produk (Kodra & Purwarianti, 2013).

2.2 *Twitter*

Twitter merupakan media sosial bertipe *micro-blogging* (blog berukuran kecil) yang didirikan oleh Jack Dorsey pada Maret 2006 dan diluncurkan pada tahun yang

sama (Sembodo, Setiawan, & Baizal, 2016). Twitter juga dapat didefinisikan sebagai media sosial yang dapat dijadikan alat berbagi ide dan pemikiran dalam 140 karakter atau kurang (Copernicus, 2009).

Twitter memiliki sejumlah objek data yang dapat diakses oleh pengguna, salah satunya *tweet object* (*tweet*). *Tweet* merupakan teks tulisan dengan 140 karakter yang ditampilkan di halaman profil pengguna (Asur & Huberman, 2010). *Tweet* terdiri dari sejumlah entitas yang terdiri dari *id, created at*, and *text*. Selain itu, dalam *tweet* juga terdapat data objek *user* (pengguna), *entities (url, mention,* dan *hashtag*), dan *entities extended (media, tweet polls, geotweet, photo, video, animated*).

Search merupakan salah satu metode yang diizinkan Twitter untuk pihak pengembang aplikasi dan pengguna untuk terlibat atau mengakses data-data Twitter. *Search* merupakan aktivitas menggunakan API Twitter untuk mendapatkan/mengunduh data *tweets (statuses)*. *Tweet* sendiri didefinisikan sebagai teks tulisan dengan 140 karakter yang ditampilkan pada halaman pengguna (Yusra, 2013).

2.3 *Natural Language Processing*

Natural Language Processing (NLP) adalah area dari penelitian dan aplikasi yang membahas bagaimana komputer bisa digunakan untuk memahami dan memanipulasi bahasa teks alami. Penelitian tentang *NLP* bertujuan untuk mengumpulkan pengetahuan tentang bagaimana manusia memahami dan menggunakan bahasa sehingga bisa dikembangkan untuk membuat sistem komputer mengerti dan memanipulasi bahasa alami untuk melakukan tugas yang diharapkan (Vijayarani, Ilmathi, & Nithya, 2015)

Aplikasi *NLP* mencakup sejumlah bidang studi, seperti terjemahan mesin, bahasa alami pengolahan teks dan *summarization*, *user interface*, pengambilan informasi multibahasa dan lintas bahasa (CLIR), pengenalan suara, kecerdasan buatan dan sistem pakar dan sebagainya.

2.4 *Preprocessing Tweet*

Preprocessing merupakan sekumpulan tahapan untuk mempersiapkan data teks sebelum dilakukan proses lain (Mujilahwati, 2016). *Preprocessing tweet* perlu mendapat penanganan khusus karena karakteristiknya yang tidak terstruktur dan rumit (Hidayatullah & Ma'arif, Pre-processing Tasks in Indonesian Tweet Messages, 2016). Hal ini disebabkan tidak adanya regulasi baku dalam tata bahasa dalam *tweet*. Selain itu tweet juga mengandung simbol, *mention*, *hashtag*, *url* dan lainnya yang dapat menjadi *noise* untuk mendapatkan informasi (Hidayatullah & Ma'arif, Pre-processing Tasks in Indonesian Tweet Messages, 2016).

Penelitian ini menggunakan *text tweet* sebagai bagian yang dijadikan fitur dalam metode pembelajaran. *Preprocessing* terhadap data *tweet* yang dilakukan pada penelitian terdiri dari beberapa tahapan. Tahapan-tahapan tersebut antara lain *case folding, cleansing, tokenizing, convert word, stopword removal* dan *stemming,*

2.4.1 *Case Folding*

Case folding merupakan tahapan untuk mengubah seluruh kata menjadi satu bentuk *uppercase* atau *lowercase* (Hidayatullah & SN, 2014). Pada penelitian ini, tahapan *case folding* merupakan tahap awal dalam *preprocessing*. Proses akan mengubah seluruh karakter huruf/alfabet yang membentuk kata menjadi huruf kecil.

2.4.2 *Cleansing*

Cleansing merupakan proses membersihkan data *tweet*/dokumen dari kata yang tidak diperlukan untuk mengurangi *noise* (Nur & Santika, 2011). Pada penelitian ini *text tweet* yang digunakan sebagai *dataset* terdapat sejumlah entitas yang tidak diperlukan untuk proses selanjutnya. Entitas-entitas tersebut antara lain *url*, *mention*, dan *hastagh*. Selain itu, proses ini juga membersihkan karakter huruf dan tanda baca yang dinilai menjadi noise dalam *tweet*/dokumen

2.4.3 *Tokenizing*

Tokenizing adalah tahap pemotongan *string* input berdasarkan tiap kata yang menyusunnya (Pratama & Trilaksono, 2015). Proses ini berfungsi untuk membagi karakter dalam dokumen teks menjadi berupa *token* yang digunakan untuk proses selanjutnya.

2.4.4 *Convert Word*

Convert Word merupakan proses yang bertujuan untuk mengkonversi kata tak baku (Mujilahwati, 2016). Dalam *tweet* berbahasa indonesia terdapat kata yang tak sesuai ejaan (kata tak baku). Terdapat beberapa ciri unik kata tak baku yang terdapat dalam *tweet* bahasa indonesia (Hidayatullah, 2015), ciri tersebut antara lain:

1. Pengguna Twitter Indonesia terkadang memanjangkan kata dengan menambah huruf dalam kata ketika menujukkan sebuah ungkapan. Contohnya kata 'hore' yang ditulis '*horeee*', kata 'ayo' yang ditulis '*ayooo*' dan banyak kata lainnya.
2. Menghilangkan huruf untuk menyingkat kata, seperti kata tidak menjadi '*tdk*','*gak*' atau '*gk*. Atau menggabungkan beberapa kata ke dalam satu kata seperti 'dan sebagainya' menjadi '*dsb*'.
3. Menggunakan angka sebagai pembentuk kata seperti kata '*hati2*' untuk mengungkapkan kata 'hati-hati', kata '*se7*' untuk mengungkapkan kata 'setuju'.

Proses *convert word* pada penelitian ini merujuk pada penelitian '*Preprocessing Task in Indonesia Twitter Messages'* (Hidayatullah & Ma'arif, 2016). Pada penelitian tersebut proses untuk menangani kata tak baku dilakukan dengan membuat sebuah kamus *convert word* yang terdiri dari kata tak baku dan kata bakunya. Proses dilakukan dengan melibatkan kamus tersebut untuk mengubah kata tak baku.

Untuk itu, pada penelitian ini menggunakan sebuah kamus *convert word* yang terdiri dari sejumlah kata tak baku yang diperlukan dalam proses pembelajaran. Kata-kata penting tersebut berpotensi menjadi fitur. Kata yang dimaksud seperti '*byr*','*bayr*', '*b4y4r*' untuk kata kata 'bayar' dan sejumlah kata lainnya.

2.4.5 *Stopword Removal*

Terdapat sejumlah kata yang perlu dihapus untuk mengurangi dimensi ruang ketika diproses yang disebut sebagai *stopword*. Kata-kata tersebut merupakan preposisi, nominal dan kata-kata lainnya yang dinilai bukan sebuah 'kata kunci' yang penting untuk pemrosesan. Untuk itu proses pembuangan kata-kata *stopword* disebut sebagai *stopword removal* (Vijayarani, Ilmathi, & Nithya, 2015).

Pembuatan kamus *stopword* pada penelitian ini menggunakan kamus *stopword* (*stoplist*) Tala yang telah dibentuk pada penelitian '*A Study Of Stemming Effects On Information Retrieval in* Bahasa Indonesia' (Tala, 2003). Pada penelitian tersebut, *stoplist* terdiri dari kata-kata yang paling umum digunakan dalam berita-berita beberapa koran berbahasa Indonesia.

Dalam penelitian ini dilakukan penambahan *stoplist* dari kata-kata yang umum terdapat *tweet* namun tidak berpotensi menjadi fitur mewakili *tweet* tersebut. Contoh dari ketentuan tersebut seperti kata satuan (ribuan, puluhan, belasan,persen,sebuah.setiap), mata uang (rupiah), nama perusahaan *e-commerce* (bukalapak, tokopedia, blibli, bukupedia,hijup), dan kata-kata lainnya.

2.4.6 *Stemming*

Stemming adalah proses pemetaan berbagai variasi morfologi kata kedalam bentuk dasarnya/*root* (Tala, 2003). Sedangkan menurut (Asian, 2007) *stemming* merupakan proses mengurangi varian morphologi menjadi dalam satu bentuk dasar (*root*). Artinya, *stemming* adalah usaha untuk memproses berbagai variasi morfologi kata ke dalam satu bentuk baku.

Dalam penelitian ini, *stemming* yang digunakan adalah *Enhanced Confix Stripping* Stemmer (ECS *Stemmer*) yang dihasilkan dari penelitian '*Enhanced Confix Stripping Stemmer and Ants Alghoritm for Clasifyin News Document In Indonesian Lenguage*' (Arifin, Mahendra, & Ciptaningtyas, 2009). *Stemming ECS* merupakan perbaikan dari *stemming* C*onfix Stripping Stemmer* (*CS Stemmer*) oleh Jelita Asian terhadap dokumen bahasa Indonesia.

Baik *ECS* maupun *CS stemmer* merupakan hasil perbaikan dan penambahan aturan *stemming* Nazief Andriani dan *stemming* Arifin Setiono. Adapun perbaikan yang dilakukan *CS Stemmer* antara lain :

1. Penggunaan kamus kata dasar yang lebih lengkap
2. Melakukan modifikasi dan penambahan aturan terhadap *stemming nazief-andriani*
3. Menambahkan aturan *stemming* terhadap kata ulang. Caranya, adalah dengan melakukan pemisahan menjadi dua sub-kata, yakni sub-kata sebelum dan sesudah tanda penghubung "-". Setelah dilakukan pemisahan, masing-masing subkata mengalami proses *stemming*. Apabila *stemming* memberikan kata dasar yang sama, maka *output* kata dasarnya adalah hasil *stemming* tersebut. Namun apabila hasil *stemming* 2 sub-kata ini berbeda, maka dapat disimpulkan bahwa input adalah kata ulang semu, dan tidak memiliki bentuk kata dasar lagi
4. Menambahkan proses pengecekan *rulePrecedence* dalam tahapan dalam proses *stemming*. Proses pengecekan *rulePrecedence* akan menentukan proses *stemming* akan melakukan penghilangan akhiran atau awalan dahulu.

Namun terdapat sejumlah kata yang tak mampu ditangani dengan metode ini. Untuk itu diperlukan metode *ECS* untuk memperbaiki kelemahan tersebut. Adapun perbaikan yang dilakukan antara lain :

1. Melakukan modifikasi dan penambahan aturan *stemming*
2. Menambahkan suatu algoritma tambahan untuk mengatasi kesalahan pemenggalan akhiran yang seharusnya tidak dilakukan. Algoritma ini disebut *loop* Pengembalian Akhiran, dan dilakukan apabila proses *recoding* gagal.

Adapun Algoritma *loop* Pengembalian Akhiran dideskripsikan sebagai berikut:

1. Kembalikan seluruh awalan yang telah dihilangkan sebelumnya, sehingga menghasilkan model kata seperti berikut: [DP+[DP+[DP]]] + Kata Dasar. Pemenggalan awalan dilanjutkan dengan proses pencarian di

kamus kemudian dilakukan pada kata yang telah dikembalikan menjadi model tersebut.

2. Kembalikan akhiran sesuai dengan urutan model pada bahasa Indonesia. Ini berarti bahwa pengembalian dimulai dari DS ("-i", "-kan", "-an"), lalu PP("-ku", "-mu", "-nya"), dan terakhir adalah P ("-lah", "-kah", "-tah", "-pun"). Untuk setiap pengembalian, lakukan langkah 3 hingga 5. Khusus untuk akhiran "-kan", pengembalian pertama dimulai dengan "k", baru kemudian dilanjutkan dengan "an".
3. Lakukan pengecekan di kamus kata dasar. Apabila ditemukan, proses dihentikan. Apabila gagal, maka lakukan proses pemenggalan awalan berdasarkan aturan *confix stripping.*
4. Lakukan *recoding* apabila diperlukan.
5. Apabila pengecekan di kamus kata dasar tetap gagal setelah *recoding*, maka awalan-awalan yang telah dihilangkan dikembalikan lagi.

. *Stemming ECS* menganalisa setiap kata/*term* dengan mengikuti format penulisan bahasa indonesia yang mengandung imbuhan (*affix*). Dalam *ECS* mengenal imbuhan berupa awalan (*prefix*) akhiran (*suffix*), sisipan (*infix*) dan awalan akhiran (*confix*). Sehinga format dasar yang digunakan dalam membuang imbuhan adalah sebagai berikut :

DP + DP + DP + root word + DS + PP + P

DP : *Derivation Prefix*

Root word : Kata Dasar

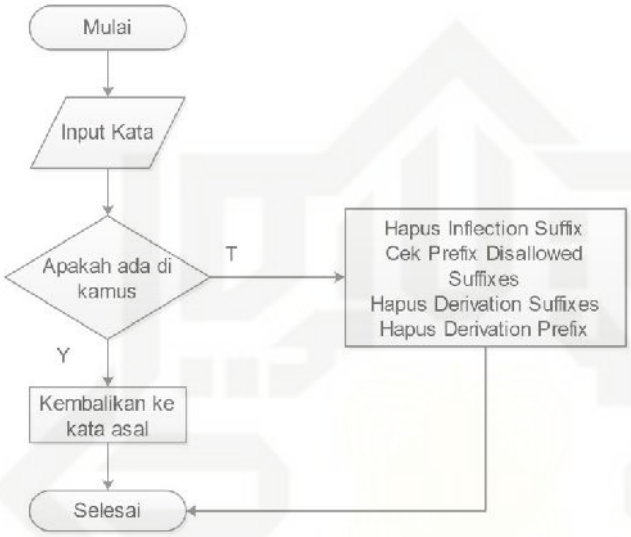
DS : *Derivation Suffix*

PP : *Possessive Pronoun (Inflection)* [ku,mu,nya]

P : Particle (*Inflection*) [lah,kah,].

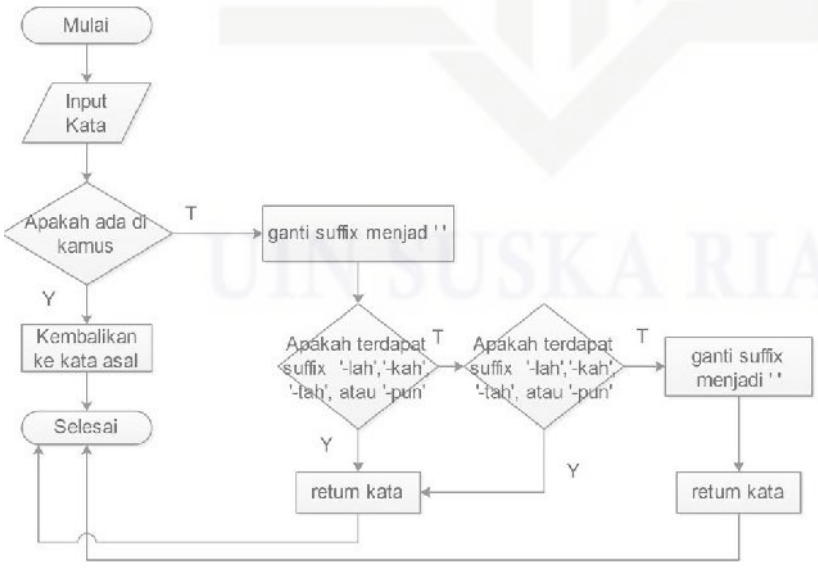
Dalam penelitian klasifikasi minat pengguna Twitter berdasarkan tweet menggunakan Support Vector Machine (SVM) (Yusra, 2013), alur proses *stemming* dijelaskan dalam tahapan berikut :

1. Cari kata yang akan di *stem* dalam kamus kata dasar. Jika ditemukan maka diasumsikan kata adalah kata dasar. Maka algoritma berhenti.



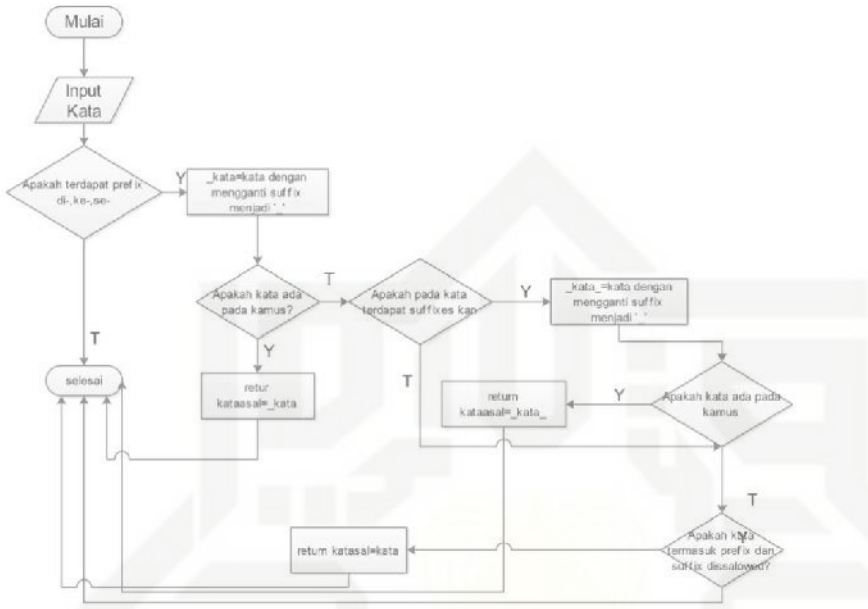
Gambar I Flowchart Pengecekan Kata di Kamus (Yusra, 2013)

2. Hapus *Inflection Suffix*



Gambar II Flowchart Hapus Inflection Suffix (Yusra, 2013)

3. Cek Kombinasi Awalan dan Akhiran yang dibolehkan
4. Hapus *Derivation Suffix* (“-i”, “-an” atau “-kan”)



Gambar III Flowchart Hapus Derivation Suffix (Yusra, 2013)

5. Hapus *derivation prefix*, yaitu : “di-”, “ke-”, “se-”, “te-”, “be-”, “me-”, “pe-”, “memper-“,

2.5 Ekstraksi Fitur dan Pembobotan

Ekstraksi fitur dapat digunakan untuk mendapatkan fitur yang digunakan dalam proses pembelajaran. Ekstraksi fitur dapat dilakukan dengan melakukan pengamatan terhadap distribusi frekuensi kemunculan kata dan jumlah fitur. Nilai *threshold* terbaik adalah titik dimana frekuensi kemunculan kata dan jumlah fitur mulai konstan (Pratama & Trilaksono, 2015).

Pada penelitian ini proses ekstraksi fitur dilakukan dengan melakukan perangkingan terhadap seluruh *term* berdasarkan bobot. Adapun metode pembobotan yang digunakan antara lain :

2.5.1 *Document Frequency*

Document Frequency (DF) adalah jumlah dokumen yang mengandung suatu *term* tertentu. *Document Frequency* merupakan metode *feature selection* yang paling sederhana dengan waktu komputasi yang rendah (Yiming & Pedersen, 1997).

2.5.2 *Term Frequency*

Term Frequency (TF) merupakan salah satu metode untuk menghitung bobot tiap *term* dalam teks. Dalam metode ini tiap *term* diasumsikan memiliki nilai kepentingan yang sebanding dengan jumlah kemunculan *term* tersebut pada teks (Hall & Smith, 1999). Bobot sebuah *term* t pada sebuah teks dirumuskan dalam persamaan berikut, Dimana TF_{dt} adalah *term frequency* dari *term* t diteks d:

$$\mathbf{W_{dt} = TF_{dt}} \tag{2.1}$$

2.5.3 *Inverse Document Frequency* (IDF)

Merupakan metode untuk menghitung kemunculan *term* dalam keseluruhan koleksi teks . Dalam hal ini, *term* yang jarang muncul pada koleksi keseluruhan *term* dinilai lebih berharga. Nilai kepentingan tiap *term* diasumsikan berbanding terbalik dengan jumlah teks yang mengandung *term* tersebut (Hall & Smith, 1999). Nilai IDF sebuah *term* t dirumuskan dalam persamaan berikut :

$$\mathbf{IDF_t = \log(\frac{n}{df})} \tag{2.2}$$

Di mana N adalah total jumlah teks / dokumen pada koleksi dan DF(t) adalah jumlah dokumen yang mengandung *term* t.

2.5.4 TFIDF

Metode TFIDF merupakan metode pembobotan yang paling umum digunakan untuk menggambarkan dokumen ke dalam model ruang vektor (Soucy & Mineau, 2005). Metode ini akan menghitung nilai *TF* dan *IDF* pada setiap token (*term*) di setiap dokumen. Metode ini akan menghitung bobot setiap token t di dokumen dengan rumus :

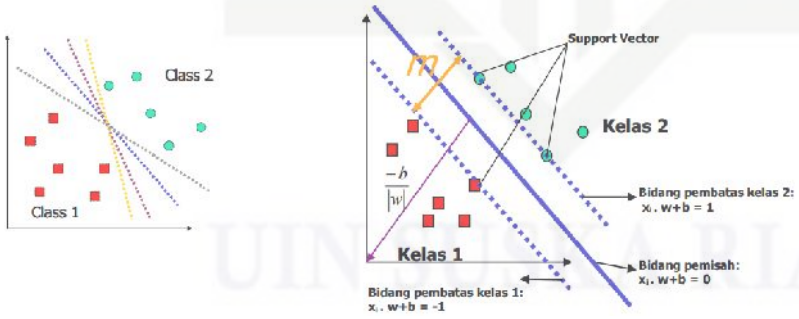
$$\mathbf{W = TF_{dt} IDF_{dt}} \tag{2.3}$$

2.6 SVM *(Support Vector Machine)*

SVM *(Support Vector Machine)* pertama kali diperkenalkan oleh Vepnik pada tahun 1992 sebagai rangkaian harmonis dari konsep komputasi ungulan yang sudah ada puluhan tahun sebelumnya (Nugroho, Witarto, & Handoko., 2003). Dalam klasifikasi, SVM bertujuan untuk menghasilkan model yang dapat memprediksi nilai target data *test* hanya melalui atribut data *test* (Chih-Wei Hsu, 2016)

SVM adalah sistem pembelajaran yang menggunakan ruang hipotesis berupa fungsi-fungsi linier dalam sebuah ruang fitur (*feature space*) berdimensi tinggi, dilatih dengan algoritma pembelajaran yang didasarkan pada teori optimasi dengan mengimplementasikan *learning* bias yang berasal dari teori pembelajaran statistik.

SVM dapat bekerja pada data linier maupun nonlinier (Sembiring, 2007). Dimaksud dengan linier artinya data dapat dipisah dengan garis pemisah secara linier. Misalkan { }n x ,..., x 1 adalah dataset dan ∈{ } +1,−1 i y adalah label kelas dari data xi. Pada gambar II-1 dapat dilihat berbagai alternatif bidang pemisah yang dapat memisahkan semua data set sesuai dengan kelasnya. Namun, bidang pemisah terbaik tidak hanya dapat memisahkan data tetapi juga memiliki margin paling besar. Adapun data yang berada pada bidang pembatas disebut *support vector*.



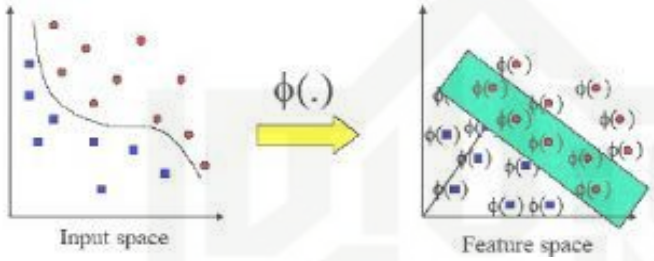
Gambar IV Alternatif bidang pemisah (kiri) dan bidang pemisah terbaik dengan margin (m) terbesar (kanan). (Sembiring, 2007)

2.6.1 SVM Non-Linier

Untuk mengklasifikasikan data yang tidak dapat dipisahkan secara linier ada cara yang dapat dilakukan yaitu dengan penambahan variabel ξi (i ≥ ∀i ξ 0, : ξ i = 0 jika

i x diklasifikasikan dengan benar) menjadi xi .w + b ≥1-ξi untuk kelas 1 dan xi .w + b ≤ −1+ξi untuk kelas 2. Pencarian bidang pemisah terbaik dengan dengan penambahan variabel ξ i sering juga disebut soft margin *hyperplane*.

Selain itu dapat juga dilakukan dengan mentransformasikan data ke dalam dimensi ruang fitur (*feature space*) sehingga dapat dipisahkan secara linier pada *feature space*.



Gambar V Transformasi dari vektor input ke *feature space.* (Sembiring, 2007)

Caranya, data dipetakan dengan menggunakan fungsi pemetaan (transformasi) (k x →φ x) kedalam *feature space* sehingga terdapat bidang pemisah yang dapat memisahkan data sesuai dengan kelasnya.

Feature space dalam prakteknya biasanya memiliki dimensi yang lebih tinggi dari vektor input (*input space*). Hal ini mengakibatkan komputasi pada *feature space* mungkin sangat besar, karena ada kemungkinan *feature space* dapat memiliki jumlah *feature* yang tidak terhingga. Selain itu, sulit mengetahui fungsi transformasi yang tepat. Untuk mengatasi masalah ini, pada SVM digunakan "*kernel trick*".

Syarat sebuah fungsi untuk menjadi fungsi *kernel* jika memenuhi Teorema Mercer yang menyatakan bahwa matriks *kernel* yang dihasilkan harus bersifat *positive semi-definite* (Sembiring, 2007). Fungsi kernel yang umum digunakan antara lain:

a. *Kernel* Linier

$$K(x_{i,}\, x) = x_i^T\, x \qquad \textbf{(2.4)}$$

b. *Pilynomial Kernel*

$$K(x_{i,}\, x) = (\gamma .\, x_i^T\, x + r)^p \;, \gamma > 0 \qquad \textbf{(2.5)}$$

c. *Radial Basis Fuction*

$$K(x_i, x)=\exp(-\gamma |x_i - x|^2)\ \gamma \quad \textbf{(2.6)}$$

d. *Sigmoid Kernel*

$$K(x_i, x)=\tanh(\gamma x_i^T x + r) \quad \textbf{(2.7)}$$

Diantara *kernel* lainnya, umumnya *kernel* RBF lebih sering digunakan dan menjadi pilihan utama dalam menyelesaikan kasus-kasus di SVM non-linier (Chih-Wei Hsu, 2016). Tidak seperti kernel linier, *kernel* ini digunakan untuk kasus non-linier. Artinya *RBF* dapat menangani hubungan antara label dan kelas atribut pada kasus non-linier. Beberapa parameter pada *kernel* RBF juga memiliki kinerja yang sama seperti titik parameter (C, γ) pada *kernel linier*. Selain itu *kernel sigmoid* mempunyai karakteritik yang hampir sama dengan RBF.

Alasan lainnya adalah, *polynimial kernel* memiliki lebih banyak *hyperparameters* daripada RBF. Jumlah *hyperparameters* yang mempengaruhi kompleksitas pemilihan model. Pada Penelitian Klasifikasi Penerima Program Beras Miskin (Raskin) di Kabupaten Wonosobo dengan Metode *Support Vector Machine* (SVM) dengan Menggunakan LibSVM juga menunjukkan *kernel* RBF menghasilkan kinerja yang lebih baik daripada keempat *kernel* lainnya (Pamuji, Safitri, & Prahutama, 2015)

2.6.2 *Cross Validation* dan *Grid Seacrh*

Dalam *kernel* RBF ada dua parameter yaitu C dan γ yang nilainya perlu diketahui. Hal ini dilakukan untuk mendapatkan parameter terbaik yang nantinya dapat menghasilkan akurasi yang tinggi dalam pelatihan. Proses ini dikenal dengan sebutan *grid-search*.

Pada umumnya proses ini dilakukan dengan *cross validation.* Menurut (Tang & Liu, 2009) *cross validation* adalah metode statistik untuk mengevalusasi dan membandingkan algoritma pembelajaran dengan membagi data menjadi dua bagian, data *training* dan data *testing*. Semua data yang dikelompokkan kedalam dua bagian tersebut secara bergantian akan digilir kedalam bagian lainnya secara berurut.

Umumnya, teknik yang digunakan dalam proses ini adalah *k-fold cross validation.* Dalam *k-fold cross-validation*, yang disebut juga dengan *rotation estimation*,

dataset yang utuh dipecah secara random menjadi 'k' subset dengan ukuran yang hampir sama dan saling eksklusif satu sama lain (Sembiring, 2007). Model dalam '*classification*' dilatih dan diuji sebanyak 'k' kali. Setiap kali pelatihan semua dilatih pada semua *fold* kecuali hanya satu *fold* saja yang disisakan untuk pengujian. Secara umum dalam *data minning* dan *machine learning* jumlah k yang digunakan sebanyak 10-*fold*-validation. (Yusra, 2013)

2.6.3 *Decomposition Method*

Decomposition Method adalah metode untuk menyelesaikan permasalahan optimasi dengan jumlah *dataset* yang besar. Metode ini bekerja dengan prinsip '*working set'*. Metode ini hanya mengubah beberapa multiplier α i dalam jumlah tertentu pada setiap iterasi, sementara nilai yang lain bernilai tetap. *Working set* merupakan kumpulan variabel yang sedang dioptimasi pada *current iteration* (Sembiring, 2007).

Salah satu teknik implementasi yang dapat digunakan adalah dengan menggunakan LibSVM (Pamuji, Safitri, & Prahutama, 2015). Adalah sebuah paket program *SVM* yang dikembangkan oleh Chi Chung Cang dan Chi Jen Li yang memiliki *wrapper* untuk bahasa PHP berupa PHP *extension* dengan nama php-svm (Abdiansyah & Wardoyo, 2015). LibSVM mendukung tiga fungsi yaitu SVC (*Support Vector Classification* - kelas biner dan kelas multi) yang dapat digunakan untuk tugas klasifikasi, SVR (*Support Regresi Vektor*), digunakan untuk tugas regresi, dan *OneClass SVM*, yang digunakan untuk estimasi distribusi. Menurut (Chih-Wei Hsu, 2016), penggunaan LibSVM mengikuti dua tahapan. Tahapan pertama LibSVM melakukan pelatihan terhadap *dataset* untuk mendapatkan sebuah model. LibSVM kemudian menggunakan model yang telah didapatkan untuk memprediksi informasi dalam sebuat *dataset* pelatihan

2.6.4 Akurasi Klasifikasi

Akurasi klasifikasi dapat menjadi evaluasi hasil model pembelajaran (Muis & Affandes, 2015). Nilai akurasi dapat dirumuskan sebagai berikut :

$$Akurasi = \frac{Jumlah\ Klasifikasi\ benar}{Jumlah\ dokumen\ ujicoba} \text{ x } 100\% \quad \textbf{(2.8)}$$

2.7 Penelitian Terkait

Beberapa penelitian terkait pembelajaran SVM dan klasifikasi dokumen/tweet yang telah dilakukan sebelumnya dapat dilihat pada tabel berikut :

Tabel 1 Penelitian Terkait

No	Penulis	Tahun	Judul	Dipublikasi Pada	Kesimpulan
1	Abdiansyah A dan Retantyo Wardoyo	2015	*Time Complexity of Support Vector Machines* (SVM) *in* LibSVM	*International Journal of Computer Aplication (09875-887)*	Penelitian ini melakukan pengujian *running time* terhadap dua bahasa pemrograman C++ dan Java. Hasilnya, kinerja klasifikasi SVM menghasilkan *running time* lebih cepat dengan bahasa pemrograman C++ dibanding Java.
2	Ariadi dan Fithriasari	2015	Klasifikasi Berita Indonesia Menggunakan Metode *Naive Besian Clasification* dan *Support Vector Machine* dengan *Config Stripping*	Jurnal Sains dan Seni Vol 4, No. 2 (2015) 2337-3520 (2301-928X Print)	Penelitian ini dilakukan untuk membandingkan hasil klasifikasi dari dua metode pembelajaran dalam klasifikasi kategori berita. Adapun metode pembelajaran yang digunakan adalah *Naive Bayes Classification* (NBC) dan *Support Vector*

No	Penulis	Tahun	Judul	Dipublikasi Pada	Kesimpulan
			Stemmer		*Machine* (SVM). Hasilnya SVM menghasilkan *nilai akurasi, precission, recall dan F-measure* yang lebih tinggi dibanding NBC.
3	Hidayatullah dan Azhari	2014	Analisis Sentimen dan Klasifikasi Kategori Terhadap Tokoh Publik Pada Twitter.	Seminar Nasional Informatika (SeminasIF) UPN "Veteran"Yogyakarta,12 Agustus 2014. ISSN 1979-2328	Penelitian ini dilakukan untuk membangun model klasifikasi berdasarkan kategori dan sentimen terhadap tokoh publik. Hasil evaluasi klasifikasi, dua metode yang dibandingkan dalam penelitian ini, yaitu NBC dan SVM menghasilkan akurasi terbaik terdapat metode SVM *kernel* RBF
4	Imelda A.Muis,Muhammad Affandes	2015	Penerapan Metode *Support Vector Machine* (SVM)	Jurnal Sains, Teknologi & Industri, Vol.12, No.2, Juni 2015, pp. 189-197 ISSN	Metode SVM dengan *kernel* RBF berhasil menghasilkan akurasi yang saat tinggi pada klasifiasi *tweet* iklan dan non iklan.

No	Penulis	Tahun	Judul	Dipublikasi Pada	Kesimpulan
			Menggunakan *Kernel Radial Basis Function* (RBF) Pada Klasifikasi *Tweet*	1693-2390 print/ISSN 2407-0939 online	Klasifikasi dengan pemilihan fitur menghasilkan nilai akurasi lebih tinggi sebesar 99,12% dibanding yang tidak dilakukan pemilihan fitur
5	Kodra, Leila Masayu dan Ayu Purwarianti		Ekstraksi Informasi Transaksi *Online* Pada Twitter	Jurnal Cybermatika, Volume 1 (2013), Issue 1, Artikel 4	Penelitian ini menjelaskan tentang aplikasi Safe-f yang melakukan klasifikasi terhadap *tweet e-commerce* dengan menggunakan pendekatan klasifikasi untuk tahapan filter dan ekstraksi.
6	Pamuji Yogi Setio, Diah Safitri, Alan Prahutama	2015	Klasifikasi Penerima Program Beras Miskin (Raskin) di Kabupaten Wonosobo dengan Metode *Support Vector*	Jurnal Gaussian, Volume 4, Nomor 4, Tahun 2015	Penelitian ini melakukan penelitian kemampuan SVM dalam mengklasifikasi data BPS untuk penerima beras miskin (raskin) di Wonosobo. Penelitian ini berupaya memberikan solusi klasifikasi dengan

No	Penulis	Tahun	Judul	Dipublikasi Pada	Kesimpulan
			Machine (SVM) dengan Menggunakan LibSVM		membandingkan keempat *kernel* non linier SVM. Dari hasil penelitian ditemukan bahwa *kernel* RBF berhasil menghasilkan akurasi terbaik sebesar 83,1933% persen dibanding *kernel* lainnya
7	Pratama, Enda Esyuda, Bambang Ryanto Trilaksono	2015	Klasifikasi Keluhan Pelangan Berdasarkan *Tweet* Dengan Menggunakan Metode *Support Vector Machine* (SVM)	Jurnal Edukasi dan Penelitian Informatika (JEPIN) Vol 1, No.2,2015	Aplikasi *tools* yang dibangun dengan metode SVM pada penelitian ini berhasil mengklasifikasi *tweet* keluhan/non-keluhan dari akun *@SpeedyTelkomsel*. Akurasi terbaik dalam ekstraksi fitur bentuk keluhan dihasilkan oleh metode *TF*. Sedangkan jenis keluhan dihasilkan oleh *Chi-Square*. Gabungan *TF-IDF,chi squared* dan menghasilkan akurasi tertinggi dalam hal jenis keluhan dan

No	Penulis	Tahun	Judul	Dipublikasi Pada	Kesimpulan
					bentuk keluhan. Sedangkan hasil evaluasi klasifikasi, terdapat angka 83.67%, 83.33%,dan 83.29% untuk *precision, recall*, dan *f-measure*
8	Yusra, Dhita Olivita, Yelli Fitriani	2016	Perbandingan Klasifikasi Tugas Akhir Mahasiswa Jurusan Tekhnik Informatika Menggunakan *Metode Naive Bayes Classifier* dan *K-Nearest Neighbour*	Jurnal Sains, Teknologi & Industri, Vol.14, No.1, Desember 2016, pp. 879-85 ISSN 1693-2390 print/ISSN 2407-0939 online	Penelitian ini melakukan perbandingan metode *Naive Bayes Classifier* dan *K-Nearest Neighbor* dengan mengimplemnetasikan menggunakan *tools weka*.