BAB II

TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1. Tinjauan Pustaka

Penelitian tentang pemanfaatan metode komputasi untuk pengoreksian jawaban dengan bahasa natural sudah dilakukan sejak tahun 1966. Sejak saat itu penilaian jawaban dengan bahasa natural menjadi bidang yang luas. Akhirnya penilaian dengan bahasa natural dibagi menjadi 2 bidang yaitu dengan jawaban singkat (ASJS) dan essai (*Essay Answer Grading*). Ini dikarenakan perbedaan antara kedua karakteristik tipe jawaban tersebut. Untuk penelitian mengenai konsep dan algoritma dari ASJS sendiri sampai sekarang sudah banyak dilakukan. Pada sub bab ini, penulis akan menjabarkan beberapa penelitian tersebut.

Burrows *et al.* (2015) menjelaskan mengenai perkembangan dari sistem ASJS dan sistem-sistem sejenis. Mereka meneliti 35 sistem ASJS dan membaginya menjadi lima kelompok atau era berdasarkan metodologi dan evaluasinya. Mereka menjelaskan bahwa pertanyaan dengan tipe jawaban singkat atau *short answer* merupakan tipe pertanyaan yang membutuhkan pemanggilan kembali (*recall*) dari pengetahuan yang telah didapatkan dan membutuhkan jawaban dengan bahasa natural (*natural language*). Mereka menjelaskan bahwa ASJS memiliki lima era diantaranya adalah *concept mapping, information extraction, corpus-based methods, machine learning,* dan *evaluation.* Burrows, Gurevych, & Stein membandingkan setiap sistem

dalam lima era tersebut dan menjelaskan konsep dan metode yang digunakan oleh ke-35 sistem ASJS yang diteliti. Mereka juga memaparkan komponen penting dalam ASJS diantaranya adalah data set, *natural language processing*, *model building*, *grading models*, *model evaluation*, dan keefektifan sistem. Secara keseluruhan Burrows, Gurevych, & Stein menjelaskan mengenai perkembangan ASJS dimana tren dari ASJS bergeser dari *rule-based method* menuju metode statistikal dengan bantuan *corpus-based method* atau metode *natural language processing* yang digunakan dalam sistem *machine learning.*

Mohler dan Mihalcea (2009) mencoba untuk membandingkan korelasi antara tiap metode *knowledge-based (shortest path, Leacock & Chodrow, Lesk, Wu & Palmer, Resnik, Lin, Jiang & Conrath, Hirst & St-Onge)* dan metode *corpus-based(LSA BNC, LSA Wikipedia, ESA Wikipedia, tf*idf).* Dari hasil perbandingan ini didapatkan fakta bahwa metode *knowledge-based (shortest path, Jiang & Conrath)* dan metode *corpus-based (LSA* dan *ESA)* memiliki hasil pembobotan yang sebanding namun *corpus-based* memiliki keuntungan karena kebebasan tata bahasa. Kemudian dalam metode corpus, domain corpus memiliki peran yang sangat penting dalam pembobotan dan besarnya corpus tidak berpengaruh besar dalam pembobotan ketika sebuah nilai ambang telah tercapai. Mereka juga menemukan teknik baru untuk mengintegrasikan jawaban dari peserta didik dengan kunci jawaaban. Dengan menggunakan metode yang mirip dengan teknik *pseudo-relevance feedback* dalam memperoleh informasi, mereka dapat meningkatkan kualitas sistem. (Mohler & Mihalcea, 2009).

Neilsen *et al.* (2009) menganalisa pengaruh fitur kata (leksikal) dan susunan tata bahasa (gramatikal) kepada sistem *automatic scoring.* Fitur leksikal yang digunakan diantaranya adalah probabilitas *entailment* setiap kata, fitur bertipe boolean yang akan bernilai *true* jika jawaban siswa memiliki nilai yang sama persis dengan kunci jawaban dan fitur hubungan antara proposisi kata. Fitur gramatikal meliputi penentuan jenis kata dan nilai kemiripan dari kalimat jawaban dengan kunci jawaban. *Dataset* yang digunakan pada penelitian ini kemudian dibagi menjadi empat. *Dataset* tersebut meliputi *training set, unseen answer, unseen questions,* dan *unseen modules.* Selanjutnya mereka mengevaluasi beberapa metode mesin belajar untuk menentukan nilai. Metode terbaik yang dapat mengklasifikasi *test sets* yang tercipta pada penelitian ini adalah C4.5. Hasil akurasi dari proses klasifikasi adalah 77,1% untuk *training set,* 75,5% untuk *unseen answers,* 61,7 untuk *unseen questions* dan 61,4 untuk *unseen modules.*

Gomma dan Fahmy (2012) meneliti mengenai penggunaan *string similarity* dan *corpus-based similarity* dalam pembangunan sistem ASJS. Dalam penelitiannya, mereka melakukan perhitungan bobot untuk *string-based* dan *corpus-based* secara terpisah kemudian hasil dari perhitungan bobot akan dikombinasikan hingga mendapatkan nilai korelasi maksimal 0,504. Nilai korelasi pada penelitian yang dicapai pada penelitian merupakan nilai terbaik untuk pendekatan *Bag of Words (BOW)* ketika dibandingkan dengan penelitian sebelumnya. Pada penelitiannya, penanganan *short answer* dilakukan dengan pendekatan BOW karena pendekatan ini mudah untuk

diimplementasikan dimana tidak dibutuhkannya algoritma NLP dan algoritma *supervised learning* yang kompleks. Dalam penelitiannya pembentukan model (*model building*) dilakukan dalam tiga tahap, tahap pertama adalah menghitung bobot kemiripan antara model kunci jawaban dengan jawaban peserta ujian. Tahap kedua adalah menghitung bobot kemiripan dengan metode *corpus-based* DICSO1 dan DISCO2. Tahap terakhir merupakan perhitungan bobot dengan mengkombinasikan hasil dari pembobotan di tahap pertama dan tahap kedua. Hasil terbaik dari pembobotan ini di dapat dari kombinasi metode N-gram dengan DISCO1.

Mohler *et al.* (2012) membangun sistem pengoreksian ujian otomatis pertama yang menggunakan pendekatan graph. Proses dari penentuan keputusan dibagi menjadi tiga tahap. Tahap pertama adalah pembentukan *node* dari jawaban penguji dan peserta ujian. *Node* ini terbentuk dari perhitungan nilai kesamaan leksikal, semantik, dan sintaksis teks masing - masing jawaban dan kunci jawaban. Tahap kedua adalah mencocokan nilai kesamaan yang terbentuk di setiap *node* dengan algoritma Hungarian. Tahapan terakhir adalah menggunakan *Support Vector Machines* (SVM) untuk menentukan nilai hasil. Nilai hasil SVM akan ditransformasi menggunakan model *Isotonic Regression* (IR) menjadi skala nol sampai dengan lima.

e-Examiner merupakan salah satu penelitian yang fokus terhadap fleksibilitas dan keterbukaan rancangan arsitektur perangkat lunaknya. Proses evaluasi jawaban mengaplikasikan *preprocessing* pada pemrosesan bahasa natural bersama dengan metode statistik metrik ROUGE (*Recall-Oriented Understudy for Gisting Evaluation*).

Pemrosesan bahasa natural pada jawaban dan kunci jawaban meliputi Tokenisasi, pemecahan/pemisahan kalimat, POS Tagger (struktur dan jenis kata), analisa morfologi kalimat, pencarian *stopword*, dan *token normalizer.* Untuk menentukan nilai, Gult melakukan 3 proses. Proses pertama adalah menghitung bobot nilai kesamaan antara jawaban dengan kunci jawaban dengan *Cosine similarity.* Proses kedua adalah menghitung karakteristik pada tingkat kata dengan model statistik ROUGE. Proses terakhir adalah penentuan nilai yang diberikan dengan kombinasi linier dari dua proses sebelumnya (Gütl, 2007).

Bailey dan Meurers (2008) membangun CAM (Content Assessment Module) yang menerapkan enam proses utama. Proses tersebut meliputi *input, annotation, pre-alignment filters, alignment, diagnosis, output. Input* yang diberikan berupa jawaban dan kunci jawaban. Proes annotation meliputi pemrosesan bahasa natural (*tokenization, POS Tagging, Spelling Corection,* dan lainnya). Proses *pre-alignment* merupakan proses penghilangan tanda baca dan simbol. Proses *alignment* meliputi perhitungan pada tiga tingkat, tingkat pertama adalah tingkat token (nilai kesamaan jenis kata), tingkat kedua adalah tingkat *chunk* (nilai kesamaan kata) dan tingkat terakhir adalah relasi antar kata. Metode mesin belajar yang digunakan pada sistem ini adalah TiMBL (Tilburg Memory-Based Learner).

Maurers *et al.* (2011) membangun sistem pengoreksian soal otomatis pertama dengan bahasa Jerman. Sistem yang dibangun dinamakan CoMiC-DE(Comparing Meaning in Context). Fungsi utama dari sistem ini meniru fungsi

dari CAM yang dibangun oleh (Bailey & Meurers, 2008). Untuk pemrosesan bahasa dan text CoMiC-DE menggunakan beberapa komponen. Komponen tersebut diantaranya adalah OpenNLP (untuk pendeteksi kalimat, *tokenization* dan penentuan kata benda), TreeTagger (untuk proses *lemmatization* dan penentuan struktur kata), Levenshtein Distance (untuk pengecekan ejaan kata), GermaNet (untuk menetukan hubungan kata dalam bahasa Jerman), PMI-IR (untuk menetukan nilai kesamaan kalimat), dan MaltParser(untuk menentukan tingkat ketergantungan hubungan antar kata). CoMiC-DE akan memberi keputusan (mengklasifikasikan) apakah jawaban dari peserta ujian benar atau salah (hanya benar atau salah). CoMiC-DE memerlukan beberapa atribut untuk menentukan hasil klasifikasinya. Atribut tersebut diantaranya adalah kata kunci, *token, chunk, dependency triple, token match, similarity match, type match, lemma match, synonym match,* dan *variety match.*

Madnani (2013) membangun sistem pengoreksi jawaban yang berupa rangkuman suatu paragraf. Tujuan dari pembangunan sistem ini adalah mengetahui kemampuan pemahaman siswa terhadap suatu bahan bacaan. Untuk membantu menentukan bobot nilai yang akan diberikan, Madnani mengaplikasikan beberapa fitur diantaranya : BLEU (*Bilingual Evaluation Understudy*), ROUGE, CopiedSumm, CopiedPassage, MaxCopy, FirstSent, Length, Coherence. ROUGE menghitung tingkat kesamaan dari jawaban siswa dengan paragraf yang tersedia. CopiedSumm, CopiedPassage, BLEU, dan MaxCopy mendeteksi setiap kalimat yang digunakan pada jawaban dan paragraf untuk menemukan hubungan dan kecocokannya. FirstSent akan

mendeteksi ide utama pada jawaban yang biasanya diberikan pada kalimat pertama pada suatu paragraf. Length akan membandingkan jumlah kalimat pada paragraf dan jumlah kalimat pada jawaban siswa. Coherence akan membandingkan dan menentukan seberapa baik siswa dapat merepresentasikan paragraf dalam rangkuman yang dibuatnya. Seluruh hasil dari proses diatas akan diklasifikasikan dengan model klasifikasi *logistic classifier* untuk menentukan tingkat pemahaman siswa.

Untuk menyaingi metode klasifikasi regresi logistik, Haltuf (2014) mencoba untuk menggunakan SVM pada sistem penilaian kreditor. Fungsi dari sistem yang dibangun adalah mengklasifikasikan kreditor ke kelas kreditor baik atau kreditor buruk (klasifikasi biner). Sistem ini sebelumnya menggunakan pendekatan regresi logistik dalam proses klasifikasinya. Dalam penelitiannya, Haltuf membuktikan bahwa penggunaan metode SVM dapat memberikan performa yang lebih baik daripada penggunaan regresi logistik walaupun perbaikan nilai yang diberikan tidak signifikan.

Dalam penelitian ini, penulis akan membangun *dataset* kearakteristik jawaban. Dalam pembentukan *dataset*, penulis akan menerapkan pemrosesan bahasa natural. Pemrosesan bahasa natural meliputi pemecahan paragraf menjadi kalimat, pemecahan kalimat menjadi kata, *stemming* kata, penentuan jenis kalimat, penghapusan *stopword,* penghitungan selisih kata, penentuan struktur kalimat dan penghitungan kemiripan kalimat. Dari proses diatas didapatkan empat fitur pada *dataset*. Fitur tersebut adalah selisih jumlah kata, nilai kesamaan jenis kata, nilai kesamaan kalimat, dan

nilai kesamaan struktur kalimat. Setelah *dataset* terbentuk, penulis akan menerapkan model klasifikasi SVM yang meliputi SVM linier dan SVM dengan kernel *radial basis function* (RBF). Hasil dari metode klasifikasi SVM kemudian akan dibandingkan dengan metode klasifikasi lainnya.

2.2. Landasan Teori

2.2.1. *Automatic Scoring* untuk Jawaban Singkat (ASJS)

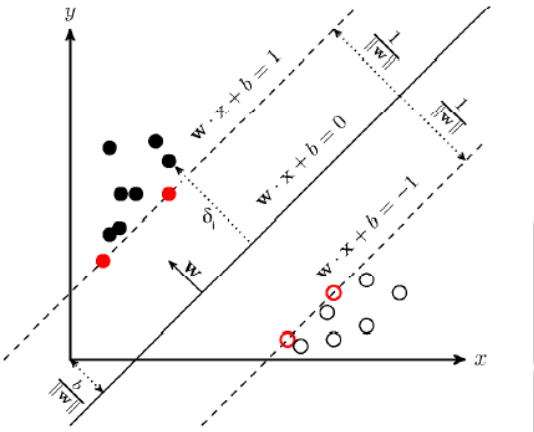
Menurut Burrows *et al.* (2015) ASJS merupakan sistem yang dapat menilai jawaban dengan bahasa natural. Pertanyaan yang digunakan haruslah pertanyaan yang objektif dan proses penilaian dibantu oleh komputer. ASJS memungkinkan sistem untuk memberikan penilaian pada setiap jawaban dengan membandingkannya dengan kunci jawaban yang telah tersedia. ASJS merupakan perluasan dari ilmu penilaian otomatis (*automatic scoring*) pada bahasa natural. Penelitian ini sudah dilakukan sejak tahun 1966. Semenjak saat itu, penelitian mengenai penilaian jawaban dengan bahasa natural semakin meluas. Teknik yang dikembangkan untuk menyelesaikan permasalahan juga semakin variatif. Hal ini menyeseuaikan dengan tipe jawaban yang akan dinilai, seperti jawaban singkat atau esai.

ASJS mempunyai 11 komponen utama yang meliputi lima proses (pembentukan *data set,* pemrosesan bahasa alami, pembentukan model, penilaian dan evaluasi model yang terbentuk) dan enam produk / hasil (penyesuaian tes, *data set,* statistik dan hasil

pemrosesan data, model penilaian, hasil penilaian dan yang terakhir adalah keefektifan sistem). ASJS sendiri memiliki lima era yang diantaranya adalah *Concept Mapping*, *Information Extraction*, *Corpus-based method*, *Machine learning*, dan *Evaluation.* Pada setiap era, ASJS memiliki konsep dan metode yang berbeda namun tetap memiliki 11 komponen ASJS.

2.2.2. Support Vector Machines (SVM)

SVM merupakan metode pembelajaran terawasi yang menghasilkan pemetaan dari fungsi *input-output* dari sekumpulan data *training.* Fungsi pemetaan ini bisa berupa fungsi klasifikasi atau fungsi regresi (Wang, 2005). SVM menggunakan ruang hipotesis berupa fungsi-fungsi linier di dalam *feature space*. SVM dilatih menggunakan algoritma pembelajaran yang didasarkan pada teori optimasi dengan mengimplementasikan *learning bias* yang berasal dari teori pembelajaran statistik (Christianini, 2000). Teori mengenai SVM sudah berkembang sejak tahun 1960an, namun pada tahun 1992 Vapnik, Boser, dan Guyon memperkenalkan kembali teori ini dan semenjak saat itu SVM berkembang dengan pesat. Gambar dibawah ini akan digunakan oleh penulis untuk mempermudah dalam memahami metode SVM.



Gambar 2.1 : Visualisasi SVM (Haltuf, 2014).

Pada gambar diatas dapat dilihat bahwa terdapat dua buah kelas data yang terpisah secara linier (kelas bulatan hitam dan kelas bulatan putih). Kedua kelas tersebut dipisahkan oleh sebuah garis yang disebut *hyperplane*, persamaan garis *hyperplane* adalah w.x + b = 0 (w adalah normal bidang dan b adalah bias atau posisi bidang relatif terhadap pusat kordinat). Sedangkan vektor yang memiliki jarak paling dekat dengan *hyperplane* disebut dengan *support vector*. SVM akan memisahkan data menggunakan *hyperplane* dengan batas antar kelas terbesar. Oleh karena itu akan dibentuk garis pembatas yang sejajar dengan *support vector* semua kelas. Persamaan yang terbentuk dari garis pembatas dari kedua kelas adalah :

untuk y = +1, maka w.x + b = 1
untuk y = -1, maka w.c + b = -1 (2.1)

Dari persamaan tersebut dapat diketahui bahwa data yang masuk kategori kelas pertama adalah data yang memiliki nilai persamaan lebih besar atau sama dengan 1 ($w.x+b \geq 1$) sedangkan data yang masuk ke kategori kelas kedua adalah data yang memiliki nilai persamaan lebih kecil atau sama dengan -1 ($w.x+b \leq -1$). Mengetahui fakta tersebut maka dari kedua garis pembatas tersebut dapat dibentuk pertidaksamaan

$$y_i\ (x_i.w + b) - 1 \geq 0 \qquad (2.2)$$

Nilai batas pada setiap bidang pembatas ke *hyperplane* adalah $\frac{1}{||w||}$, maka dapat ditentukan nilai batas gabungan adalah $\frac{1}{||w||} + \frac{1}{||w||} = \frac{2}{||w||}$. Selanjutnya nilai batas ini akan dimaksimalkan, memaksimalkan nilai batas berarti meminimalkan nilai dari ||w|| sebagai penyebut. Jika kedua bidang pembatas direpresentasikan sebagai (2.2) maka pencarian *hyperplane* dengan batas terbesar dapat dirumuskan menjadi masalah optimasi konstrain, yaitu

$$\frac{1}{2}\ |w|^2 \qquad (2.3)$$

$$\text{dengan } y_i\ (x_i.w + b) - 1 \geq 0$$

Kemudian akan dilakukan optimalisasi untuk menanggulangi *dataset* yang besar menggunakan *Largrangian Method.* Metode *largrangian* akan merubah rumusan (2.3) menjadi

$$L_P(w,b,\alpha) = \frac{1}{2}\ |w|^2 - \sum_{i=1}^{n} \alpha_i(y_i(x_i.w + b) - 1) \quad (2.4)$$

α merupakan nilai dari koefisien *largrangian* dengan nilai $\alpha_i \geq 0$. Selanjutnya persamaan diatas akan

diminimumkan terhadap w dan b sehingga $\frac{\partial}{\partial b}$ $L_P(w,b,\alpha)$ = 0 dan $\frac{\partial}{\partial w}$ $L_P(w,b,\alpha)$ = 0. Sehingga didapatkan

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i \quad (2.5)$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \quad (2.6)$$

Namun karena kemungkinan vektor w bernilai tak terhingga maka formula *largrangian* yang sebelumnya dalam bentuk *primal problem* diubah ke bentuk *dual problem* dengan mensubstitusikan persamaan (2.6) ke (2.4)

$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i . x_j \quad (2.7)$$

Dengan diketahui persamaan berikut

$$\min_{w,b} L_P = \max_{\alpha} L_D \quad (2.8)$$

Maka rumus untuk mencari bidang pemisah atau *hyperplane* terbaik adalah

$$\max_{\alpha} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i . x_j \right) \quad (2.9)$$

dengan

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

Rumus diatas menghasilkan vektor α yang akan digunakan untuk menghitung nilai dari w menggunakan rumus (2.5). Formulasi (2.9) merupakan permasalahan *quadratic programming*, yang menyebabkan α_i akan selalu memiliki nilai. Karena nilai α_i selalu ada / ditemukan

maka klasifikasi dari data pengujian x dapat ditentukan dengan rumus

$$f(x) = sgn(\sum_{i=1}^{n} \alpha_i y_i x_i . x - b) \quad (2.10)$$

Rumus diatas hanya akan mengklasifikasi berdasarkan *support vector* yang ada. Oleh karena itu, metode ini dinamakan *support vector machines.* (2.10) hanya bisa mengklasifikasi kelas yang dapat dipisahkan secara linier. Akan tetapi, SVM memiliki kelebihan dibandingkan metode klasifikasi yang lain karena pasti akan memisahkan data dengan *hyperplane* terbesar.

2.2.3. *Kernel* pada SVM

Penggunaan kernel pada SVM bertujuan untuk mengklasifikasi data yang tidak bisa terklasifikasi secara linier. Penggunaan kernel ini dikarenakan oleh data yang ada di kehidupan sehari-hari tidak semuanya dapat dipisahkan secara linier. Penggunaan kernel pada SVM menyebabkan meningkatnya kompleksitas perhitungan untuk klasifikasi.

Ide dari kernel ini adalah memetakan vektor input dari *original input space* ke *feature space* dengan dimensi tinggi dengan beberapa fungsi pemetaan non-linier. *Feature space* akan menyebabkan perubahan pada rumus (2.8) dimana *dot product* x_i . x_j akan dikonversi menjadi vektor transformasi $\boldsymbol{\phi}(x_i)$. (x_j).

Transformasi $\boldsymbol{\phi}(x)$ akan sangat kompleks dimana hasil yang diperoleh dapat menghasilkan dimensi yang tinggi bahkan dimensi yang tak berhingga. Oleh karena

itu, maka dikembangkanlah ide untuk membuat fungsi kernel yang didefenisikan sebagai *dot product* dari dua vektor transformasi.

Diketahui bahwa : $\mathbb{R}^n \rightarrow \mathbb{R}^m$ dengan m ≥ n, maka akan terbentuk fungsi

$$K(x,z) = \boldsymbol{\phi}(x).\ (z) \qquad (2.11)$$

Dimana $x,z \in \mathbb{R}^n$ disebut sebagai fungsi kernel. Selanjutnya dari rumus diatas, dapat disubstitusikan terhadap rumus algoritma pelatihan (2.9) menjadi

$$\underset{\alpha}{\alpha = argmax}\ (\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i.x_j))$$

dengan (2.12)

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

Sedangkan fungsi klasifikasinya dapat didefenisikan sebagai

$$f(x) = sgn(\sum_{i=1}^{n} \alpha_i y_i K(x_i.x) - b) \qquad (2.13)$$

Adapun jenis *kernel* pada SVM dapat dilihat pada tabel berikut

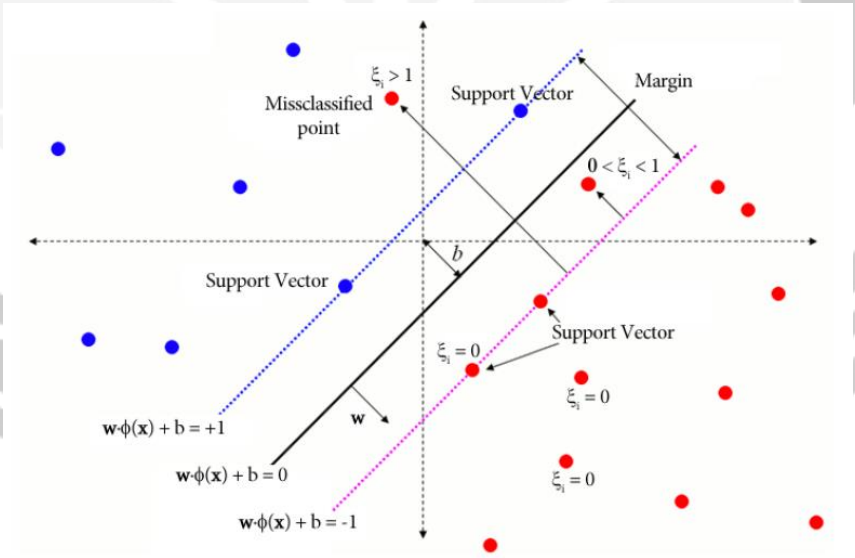
Tabel 2.1 : Kernel Pada SVM

Nama *kernel*	Fungsi *kernel* K(x,z)
Linier	$x.z$
Polynomial of degree d	$(x.z)d$
Polynomial of degree up to d	$(x.z + c)d$
Gaussian RBF	$exp\left\{-\frac{\|\|x-z\|\|}{2\sigma^2}\right\}$
Tangen Hiperbolik	$\tanh(\sigma(x.y) + c)$

Invers multi kuadratik	$\frac{1}{\sqrt{\|\|x-y\|\|^2+c^2}}$
Additive	$\sum_{i=1}^{n} K_i(xi, yi)$

2.2.4. Soft Margin Classifier

Algoritma ini digunakan untuk mengatasi permasalahan pada *dataset* yang tidak dapat diklasifikasikan atau *non-sparable data.* Ide dari algoritma ini adalah melakukan generalisasi pada rumus yang sudah ada dengan menambahkan variabel yang menjadi penanda posisi vektor yang salah pada *feature space*.



Gambar 2.2 : Visualisasi *Soft Margin Classifier* (Haltuf, 2014).

Dengan menambahkan variabel ξ maka pertidaksamaan pada (2.2) akan menjadi

$$y_i \ (x_i.w + b) \geq 1 - \xi \qquad (2.14)$$

Untuk data yang berada pada kelas yang benar nilai dari variabel ξ = 0 dan untuk data yang salah variabel nilai variabel ξ > 0. Dengan adanya variabel baru ini, maka formula pencarian *hyperline* terbaik berubah menjadi

$$\min_{w,\xi,\mathrm{b}} \left(\frac{1}{2} w.w + C \sum_{i=1}^{n} \xi\mathrm{i}\right) \quad (2.15)$$

C merupakan nilai yang ditentukan oleh pengguna, nilai ini menentukan besarnya penalti akibat kesalahan klasifikasi dalam data. Selanjutnya rumus 2.12 setelah ditambahkan variabel ξ menjadi

$$\max_{\alpha} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i . x_j)\right) \quad (2.16)$$

dengan

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

perumusan pada *soft margin classifier* ini hampir sama dengan kasus pemisahan data secara linier, namun terdapat perbedaan dimana nilai dari α_i adalah $C \geq \alpha_i \geq 0$.

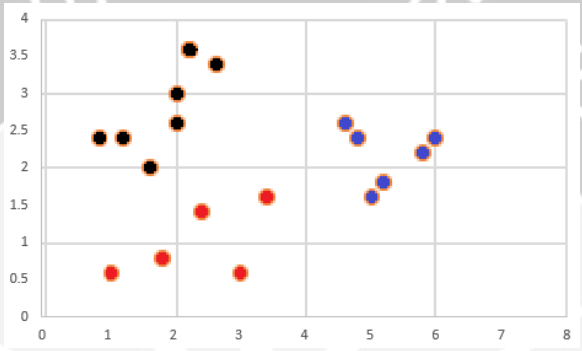
2.2.5. *Multi Class* SVM

Pada awalnya SVM memang dirancang untuk menjawab permasalahan klasifikasi biner, namun kini SVM telah dikembangkan untuk menjawab permasalahan klasifikasi non-biner. Pendekatan non-biner tersebut diantaranya adalah metode *one-against-all, one-against-one* dan *directed acyclic graph support vector machine (DAGSVM).* Untuk rumus yang biasanya digunakan

dalam *multi class* SVM adalah *soft margin classifier.* (Sembiring, 2007)

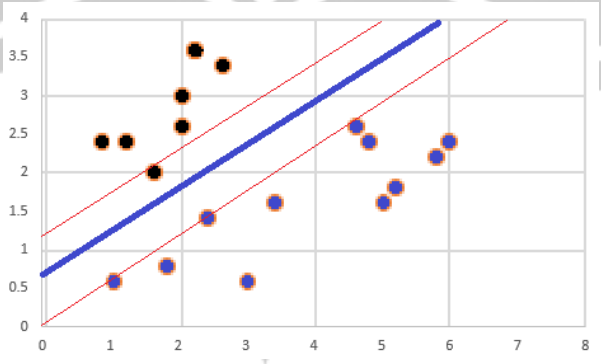
a. Metode *One-Againts-All*

Metode ini akan membandingkan setiap model klasifikasi ke-i terhadap keseluruhan data selain kelasnya. Ilustrasi dari metode ini dapat dilihat pada gambar berikut



Gambar 2.3 : Titik data dengan tiga kelas.

Misalkan terdapat tiga kelas klasifikasi, untuk menentukan *hyperplane* – nya maka akan dibentuk kelas biner dengan menggabungkan dua seperti gambar berikut



Gambar 2.4. : Visualisasi metode *one-againts-all.*

Dapat dilihat pada gambar bahwa kelas warna biru dan merah digabungkan. Dengan begitu dapat ditentukan *hyperplane* yang akan terbentuk. Hal ini juga berlaku untuk menentukan *hyperplane* untuk margin kelas yang lain.

b. Metode *One-Againts-One*

Metode ini akan memasangkan seluruh kelas dalam model, sehingga terbentuk (k(k-1))/2 buah klasifikasi biner (k adalah jumlah kelas). Misalnya terdapat tiga kelas A, B, dan C maka akan terbentuk 3 pasang klasifikasi biner yaitu A->B, A->C , dan B->C.

c. Metode DAGSVM

Metode ini hampir sama dengan metode *one-againts-one* yaitu membentuk (k(k-1))/2 klasifikasi biner. Hanya saja pada saat pengujian digunakan konsep *binary acyclic graph.* Konsep ini akan membentuk graph dimana proses evaluasi dimulai dari simpul akar kemudian bergerak ki kiri atau ke kanan tergantung nilai yang dihasilkan dari proses klasifikasi.

2.2.6. WEKA *(Waikato Environment of Knowledge Analysis)*

WEKA atau *Waikato Environment of Knowledge Analysis* merupakan perangkat lunak yang menyediakan kumpulan dari algoritma mesin belajar dan *tools* untuk *preprocessing* data. WEKA dibangun dengan bahasa Java yang kemudian didistribusikan dibawah lisensi GNU General Public. Weka menyediakan berbagai varian fungsi untuk mentransformasikan *dataset* yang kita

miliki. Dalam perangkat lunak ini kita dapat melakukan *preprocesing* data, memasukannya ke dalam skema mesin belajar, kemudian menganalisa performa dari metode yang digunakan tanpa mengetikan kode sama sekali. Perangkat lunak WEKA menyediakan semua metode standar untuk *data mining* seperti regresi, klasifikasi, *clustering*, dan analisis asosiasi. (Witten & Frank, 2005)

2.2.7. K-fold Cross Validation

K-fold Cross Validation merupakan teknik pembelajaran *dataset* yang memecah *dataset* sebanyak k subset. Satu dari subset ini akan dijadikan sebagai data testing dan k-1 subset sisanya digunakan untuk proses pembelajaran model. Proses ini dilakukan sebanyak k kali sehingga setiap subset akan menjadi data testing dari model. Proses ini akan mendapatkan k buah nilai performa dari proses pembelajaran. Semua nilai performa ini akan dicari rata-ratanya dan nilai dengan rata-rata tertinggi akan dipilih sebagai model. Keuntungan dari penggunaan *k-fold cross validation* adalah lebih efisiennya *dataset* yang terbentuk namun metode ini memiliki kelemahan karena proses komputasi yang digunakan akan lebih besar karena akan melakukan k kali proses. (Haltuf, 2014)

2.2.8. Natural Language Processing

Pemrosesan bahasa natural / *Natural language processing (NLP)* merupakan cabang ilmu dari Intelegensi Buatan dan linguistik, yang bertujuan

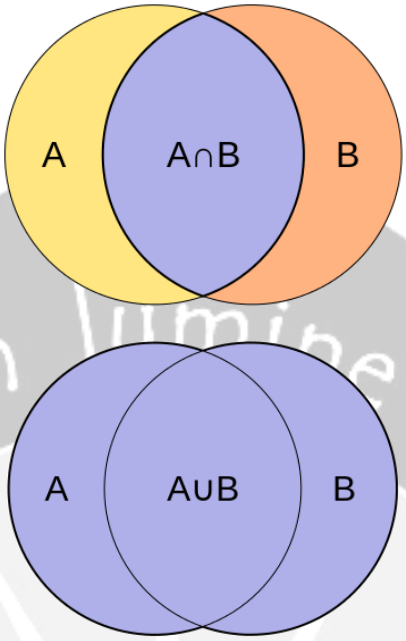
untuk membuat komputer mengerti kalimat atau kata yang dituliskan dalam bahasa manusia. Bahasa natural juga dikenal sebagai bahasa keseharian yang dibicarakan atau dituliskan manusia dengan tujuan untuk berkomunikasi. Terdapat 5 fase dari NLP diantaranya adalah *Morphological* dan *Lexical Analysis, Syntatic Analysis, Semantic Analysis, Discourse Integration,* dan *Pragmatic Analysis.* (Chopra, Prashar, & Sain, 2013)

2.2.9. Jaccard Similarity

Niwattanaku *et al.* (2013) menyebutkan bahwa *jaccard similarity* atau *jacard index* merupakan algoritma untuk menghitung kemiripan, ketidakmiripan, dan jarak antara dua buah *dataset.* Algoritma ini akan menghitung bobot kemiripan antara *dataset* dengan menghitung jumlah data yang sama kemudian membaginya dengan jumlah seluruh anggota *dataset.* Algoritma ini dapat diformulasikan sebagai berikut :

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Dimana A adalah *dataset* A dan B adalah *dataset* B dan nilai hasilnya diantara 0 sampai dengan 1 ($0 \leq J(A,B) \leq 1$). Berikut merupakan ilustrasi dari algoritma jaccard.



Gambar 2.5 : Ilustrasi algoritma jaccard.