BAB III
LANDASAN TEORI

3.1. Rekam Akademis

Rekam akademis mempunyai unsur – unsur yang dimiliki dalam sistem perkuliahan Fakultas Matematika dan Ilmu Pengetahuan Alam (FMIPA) yaitu salah satunya yaitu nilai mata kuliah. Nilai mata kuliah dalam sistem perguruan tinggi merupakan penilaian terhadap kegiatan dan kemajuan belajar mahasiswa dilakukan secara berkala yang dapat berbentuk ujian, pelaksanaan tugas, dan pengamatan dengan kriteria nilai akhir. Hasil akhir penilaian tersebut wajib dikonversikan ke dalam bentuk huruf yang memiliki sebutan, harkat, dan makna pencapaian kualifikasi (Prodi Statistika Universitas Islam Indonesia, 2017). Mata kuliah adalah satuan pembelajaran yang terdiri atas bahan kajian atau materi ajar yang dibangun untuk memenuhi capaian pembelajaran yang dirumuskan dalam Kurikulum. Sedangkan kurikulum adalah seperangkat rencana dan pengaturan mengenai capaian pembelajaran lulusan, bahan kajian, proses, dan penilaian yang digunakan sebagai pedoman penyelenggaraan Program Studi.

Mata Kuliah dapat dikelompokkan berdasarkan: Penanggung jawab pelaksanaan mata kuliah yang terdiri atas kelompok mata kuliah universitas dan kelompok mata kuliah program studi. Mata kuliah universitas adalah mata kuliah yang diterapkan untuk semua program studi Universitas Islam Indonesia dan merupakan mata kuliah wajib. Sedangkan mata kuliah program studi adalah mata kuliah yang diterapkan sesuai dengan karakter bidang keilmuan suatu program studi (Universitas Islam Indonesia, 2017). Dalam program studi statistika dibagi lagi menjadi mata kuliah inti, mata kuliah pilihan wajib dan mata kuliah pilihan bebas daftar jenis mata kuliah sesuai dengan buku panduan akademik Prodi Statistika FMIPA UII 2017/ 2018 dapat dilihat dalam lampiran 2 dan 3.

Sistem penyelenggaraan proses belajar mengajar pada program studi strata satu (S1) menggunakan Sistem Kredit Semester (SKS). Sistem Kredit Semester (SKS) adalah takaran waktu kegiatan belajar yang dibebankan pada mahasiswa per pekan per semester dalam proses pembelajaran melalui berbagai bentuk pembelajaran

atau besarnya pengakuan atas keberhasilan usaha mahasiswa dalam mengkuti kegiaran kurikuler di suatu program studi (Universitas Islam Indonesia, 2017). Dalam Sistem Kredit Semester mahasiswa diberikan kebebasan untuk menyusun rencana studi dengan memperhatikan mata kuliah yang ditawarkan, mata kuliah prasyarat dan indeks prestasi, dalam perkuliahan prodi statistika menerapkan jumlah sks antara 1, 2 atau 3 permata kuliahnya (Prayitno, 2009). Sistem penyelenggaraan proses belajar mengajar pada program studi strata satu (S1) menggunakan Sistem Kredit Semester (SKS). Dalam Sistem Kredit Semester mahasiswa diberikan kebebasan untuk menyusun rencana studi dengan memperhatikan mata kuliah yang ditawarkan, mata kuliah prasyarat dan indeks prestasi (Ndoen, et al., 2018).

Penilaian Terhadap kegiatan dan kemajuan belajar mahasiswa dilakukan secara berkala yang dapat berbentuk ujian, pelaksanaan tugas, dan pengamatan dengan kriteria nilai akhir. Hasil akhir penilaian tersebut wajib dikonversikan ke dalam bentuk huruf yang memiliki sebutan, harkat, dan makna pencapaian kualifikasi, yaitu sebagai berikut: (Prodi Statistika Universitas Islam Indonesia, 2017)

Tabel 3.1 Keterangan Nilai

A	80.00 – 100	C+	57.50 – 59.99
A-	76.25 – 79.99	C	55.00 – 57.49
A/B	72.50 – 76.24	C-	51.25 – 54.99
B+	68.75 – 72.49	C/D	47.50 – 51.24
B	65.00 – 68.74	D+	43.75 – 47.49
B-	62.50 – 64.99	D	40.00 – 43.74
B/C	60.00 – 62.49	E	< 40.00

Pada hasil akhir penilaian sebagaimana dimaksud pada peraturan Universitas Islam Indonesia No.2 Tahun 2017 tentang proses pendidikan dan peraturan pembelajaran disebutkan bahwa :

1) Nilai A dan A- yang disebut "Amat Baik", bermakna mahasiswa menunjukkan pemenuhan pencapaian pembelajaran yang unggul dan

inovatif serta keterlibatan dalam partisipasi dalam pembelajaran yang sangat baik;

2) Nilai A/B, B+, B dan B- menunjukkan prestasi pemenuhan pencapaian pembelajaran yang baik dan keterlibatan dalam aktivitas pembelajaran yang baik;
3) Nilai B/C, C+, C, dan C- yang disebut "Cukup", bermakna mahasiswa menunjukkan kecukupan pencapaian pembelajaran dan keterlibatan dalam aktivitas pembelajaran yang cukup baik;
4) Nilai C/D, D+, dan D yang disebut "Kurang", bermakna mahasiswa menunjukkan pemenuhan pencapaian pembelajaran yang rendah dan menunjukkan aktivitas pembelajaran yang rendah;
5) Nilai E yang disebut "Sangat Kurang", bermakna mahasiswa tidak dapat menunjukkan pemenuhan pencapaian pembelajaran darr/ atau tidak menunjukkan ativitas pembelajaran yang mencukupi untuk dinilai; dan
6) Nilai F yang disebut "Tidak Memenuhi Syarat untuk Dinilai", bermakna mahasiswa tidak menunjukkan aktivitas pembelajaran yang memadai. (Universitas Islam Indonesia, 2017)

3.2. Klasifikasi

Data mining atau lebih di kenal juga dengan sebutan *knowledge discovery in databases* (KDD). Data mining merupakan salah satu cara yang digunakan untuk mendapatkan pengetahuan baru dengan memanfaatkan jumlah data yang sangat besar. Beberapa teknik telah dikembangkan dan diimplementasikan untuk mengekstrak pengetahuan dan informasi untuk menemukan pola pengetahuan yang mungkin berguna untuk pengambilan keputusan. Teknik-teknik yang digunakan untuk pengekstrakan pengetahuan dalam data mining adalah pengenalan pola, clustering, asosiasi, prediksi dan klasifikasi. Klasifikasi adalah pemprosesan untuk menemukan sebuah model atau fungsi yang menjelaskan dan mencirikan konsep atau kelas data, untuk kepentingan tertentu (Defiyanti & Jajuli, 2015).

Diantara skenario pembelajaran klasifikasi, terdapat dua skenario yaitu *supervised* learning atau pembelajaran terawasi dan *unsupervised learning* atau pembelajaran tidak terawasi (Yarowsky, 1995).

Dalam *supervised learning* atau pembelajaran yang diawasi, peneliti memberikan beberapa objek penelitian yang telah mempunyai label untuk dilatih yang disebut sebagai data *training*. Tujuan *supervised learning* adalah untuk mendapatkan prediksi hubungan antara suatu atau beberapa objek kedalam labelnya. Misalnya untuk menemukan *classifier* atau kelompok yang memetakan objek ke label (Yarowsky, 1995).

Dalam *unsupervised learning* atau pembelajaran yang tidak diawasi, peneliti memberikan sejumlah objek sebagai data yang tidak memiliki label (*unlabeled training data*) kemudian objek hanya dikelompokkan berdasarkan karakteristik yang sama (Yarowsky, 1995).

3.3. *Support Vector Machine (SVM)*

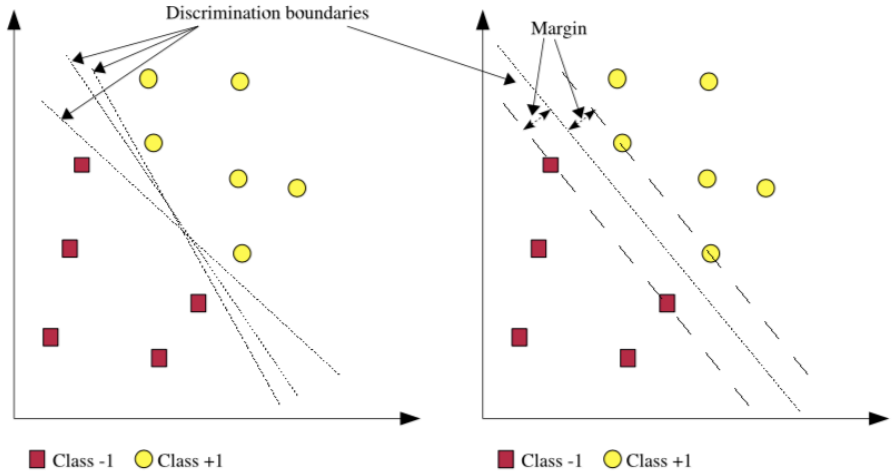
Support Vector Machine (SVM) dikembangkan oleh Boser, Guyon, dan Vapnik, pertama kali diperkenalkan pada tahun 1992 di *Annual Workshop on Computational Learning Theory*. Konsep dasar metode SVM sebenarnya merupakan gabungan atau kombinasi dari teori-teori komputasi yang telah ada pada tahun sebelumnya, seperti *marginhyperplane* (Vapnik, 1999).

Support Vector Machine (SVM) merupakan analisis klasifikasi yang menggunakan bidang pemisah (*hyperplane*) dalam melakukan klasifikasi. Pada ruang berdimensi *d*, bidang pemisah akan berdimensi *d−1*. SVM dapat menghasilkan berbagai kemungkinan bidang pemisah. Bidang pemisah terbaik yaitu yang memiliki *margin* paling besar (maximal margin hyperplane). *Margin* adalah jarak terdekat amatan data latih dengan bidang pemisah. Amatan yang memiliki jarak terdekat dengan bidang pemisah disebut *support* (James, et al., 2013).

SVM berusaha menemukan hyperplane yang terbaik pada *input space*. Prinsip dasar SVM adalah linear classifier, dan selanjutnya dikembangkan agar dapat bekerja pada problem non-linear. dengan memasukkan konsep kernel trick pada ruang kerja berdimensi tinggi. SVM dibagi menjadi dua yaitu Linear Support Vector Machine dan Non-linear Support Vector Machine, berikut adalah penjelasannya: (Nugroho, 2007)

3.3.1.*Linear Support Vector Machine*

Konsep SVM dapat dijelaskan secara sederhana sebagai usaha mencari hyperplane terbaik yang berfungsi sebagai pemisah dua buah class pada input space. Pada gambar 3.1 memperlihatkan beberapa *pattern* yang merupakan anggota dari dua buah class : +1 dan –1. *Pattern* yang tergabung pada *class* –1 disimbolkan dengan warna merah (kotak), sedangkan pattern pada *class* +1, disimbolkan dengan warna kuning(lingkaran). (Nugroho, 2007)



(Sumber : Nugroho, 2007)

Gambar 3.1 SVM berusaha menemukan *hyperplane* terbaik yang memisahkan kedua class –1 dan +1

Problem klasifikasi dapat diterjemahkan dengan usaha menemukan garis (*hyperplane*) yang memisahkan antara kedua kelompok tersebut. Berbagai alternatif garis pemisah (*discrimination boundaries*) ditunjukkan pada gambar 3.1. *Hyperplane* pemisah terbaik antara kedua *class* dapat ditemukan dengan mengukur margin *hyperplane* tsb. dan mencari titik maksimalnya. *Margin* adalah jarak antara *hyperplane* tersebut dengan *pattern* terdekat dari masing-masing *class*. *Pattern* yang paling dekat ini disebut sebagai *support vector*. Garis solid pada gambar 3.1 sebelah kanan menunjukkan *hyperplane* yang terbaik, yaitu yang terletak tepat pada tengah-tengah kedua class, sedangkan titik merah dan kuning yang berada dalam

lingkaran hitam adalah *support vector*. Usaha untuk mencari lokasi *hyperplane* ini merupakan inti dari proses pembelajaran pada SVM. (Nugroho, 2007)

Tiap data dinotasikan sebagai $x_i \in R^d$, $i = 1, 2, \cdots, N$. N adalah banyaknya data. *Positive class* dinotasikan sebagai +1, dan *negative class* sebagai −1. Dengan demikian, tiap data dan label *class*-nya dinotasikan sebagai : $y_i \in \{-1, +1\}$. Diasumsikan bahwa kedua class tersebut dapat dipisahkan secara sempurna oleh *hyperplane* di *p-dimensional feature space*. *Hyperplane* tersebut didefenisikan sebagai berikut: (Nugroho, 2007)

$$w.x_i + b = 0 \tag{3.1}$$

Data *pattern* x_i yang tergolong ke dalam *negative class* adalah mereka yang memenuhi pertidaksamaan berikut : (Nugroho, 2007)

$$w.x_i + b \leq -1 \tag{3.2}$$

Adapun data x_i yang tergolong ke dalam *positive class*, adalah mereka yang memenuhi pertidaksamaan : (Nugroho, 2007)

$$w.x_i + b \geq 1 \tag{3.3}$$

Dengan:

w : vektor bobot
x : nilai masukkan atribut
b : bias

Optimal margin dihitung dengan memaksimalkan jarak antara *hyperplane* dan *pattern* terdekat. Jarak ini dirumuskan sebagai 1/‖w‖ (‖w‖ adalah norm dari *weight vector* w). Selanjutnya, masalah ini diformulasikan ke dalam *Quadratic Programming* (QP) prolem, dengan meminimalkan rumus 3.4 dibawah rumus 3.5. (Nugroho, 2007)

Minimize:

$$min_w = \tau(w) = \frac{1}{2}\|w\|^2 \quad (3.4)$$

Kedalam :

$$y_i(w.w_i + b) - 1 \geq 0, \forall_i \quad (3.5)$$

Optimisasi ini dapat diselesaikan dengan *Lagrange Multipliers, Lagrange Multipliers* adalah metode untuk menemukan harga atau nilai minimum atau maksimum relatif dari suatu fungsi: (Muñoz, et al., 2019)

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i y_i \, (w.w_i + b - 1) \quad (3.6)$$

α_i adalah *Langrange multiplier* yang berkorespondensi dengan x_i. Nilai α_i adalah nol atau positif $\alpha_i \geq 0$, untuk menyelesaikan masalah tersebut. pertama-tama meminimalkan L terhadap *w*, dan memaksimalkan *L* terhadap α_i. $\forall_i$ yaitu untuk semua anggota *i* . Dengan memodifikasi persamaan 3.6, *maximization problem* di atas dapat direpresentasikan dalam α_i, yaitu sebagai berikut: (Nugroho, 2007)

Maximize:

$$\sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j=1} \alpha_i \alpha_j y_i y_j x_i x_j \quad (3.9)$$

Subjek menjadi :

$$\alpha_i \geq 0, \sum_{i=1}^{N} \alpha_i y_i = 0 \quad (3.10)$$

Solusi dari problem ini akan menghasilkan banyak α_i dengan nilai nol. Data yang berkorespondensi dengan α_i yang tidak nol, merupakan *support vector*, yaitu data yang memiliki jarak terdekat dengan *hyperplane*. (Nugroho, 2007)

3.3.2. *Soft Margin*

Penjelasan di atas berdasarkan asumsi bahwa kedua belah *class* dapat terpisah secara sempurna oleh *hyperplane*. Akan tetapi, pada umumnya kedua buah *class*

tersebut tidak dapat terpisah secara sempurna. Hal ini menyebabkan proses optimisasi tidak dapat diselesaikan, karena tidak ada *w* dan *b* yang memenuhi pertidaksamaan 3.5. (Nugroho, 2007)

Untuk itu, pertidaksamaan 3.5 dimodifikasi dengan memasukkan *slack variable* atau variabel kesalahan ξ_i ($\xi_i \geq 0$), menjadi : (Nugroho, 2007)

$$y_i(w.w_i + b) \geq 1 - \xi_{i,} \forall_i \tag{3.9}$$

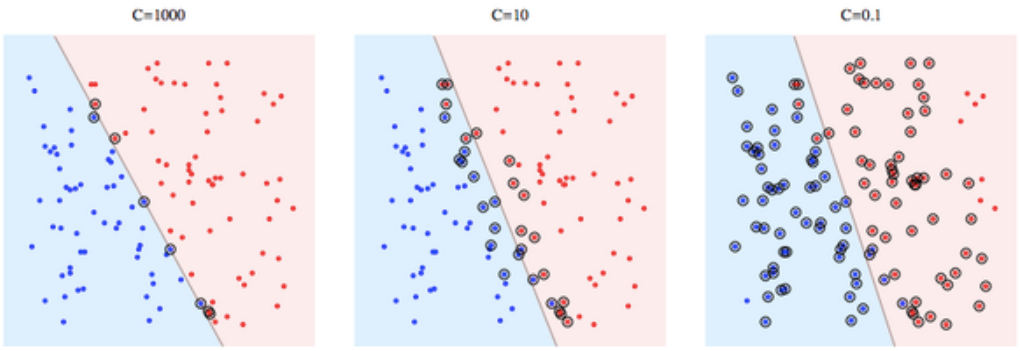
Demikian juga dengan rumus 3.4, sehingga diperoleh :

Minimize:

$$\frac{1}{2}\|w\|^2 + C \sum_{i=1}^{N} \xi_i \tag{3.10}$$

Parameter C atau *Cost* bertugas mengkontrol *tradeoff* atau keseimbangan antara *margin* and *classification error* (ξ). Semakin besar nilai C, semakin besar penalty yang dikenakan untuk tiap *classification error*. (Nugroho, 2007)

Nilai C atau *Cost* adalah seberapa banyak memberikan pinalti SVM untuk titik data dalam margin. Apabila nilai C besar, SVM akan mencoba menemukan *hyperplane* dan margin sehingga ada beberapa *pattern* dalam margin, yang memiliki model yang terlalu kompleks dengan hanya memiliki margin kecil dan bisa juga *pattern* tidak mudah dipisahkan. Serta apabila nilai C yang terlalu kecil maka akan memberikan kesalahan yang lebih tinggi pada data *training*, dan kebalikkannya maka akan memiliki margin yang besar yang mungkin memiliki model yang lebih baik dan kuat. Selain nilai C atau *Cost* ada juga nilai *gamma* yang sering dipakai dalam membentuk suatu model klasifikasi, C dan gamma adalah parameter untuk *nonlinear Support Vector Machine* dengan *Gaussian radial basis function kernel*. C adalah parameter untuk *soft margin* untuk mengontrol pengaruh pada setiap individual *pattern support vector*, C membutuhkan pemahaman tentang *slack variabel* atau variabel error seperti pada rumus 3.10, dengan *slack variabel* akan mendefinisikan seberapa banyak margin yang ditambahkan. Perhatikan ilustrasi untuk nilai C pada gambar 3.2 berikut: (Vapnik, 1999)



(Sumber : www.quora.com)

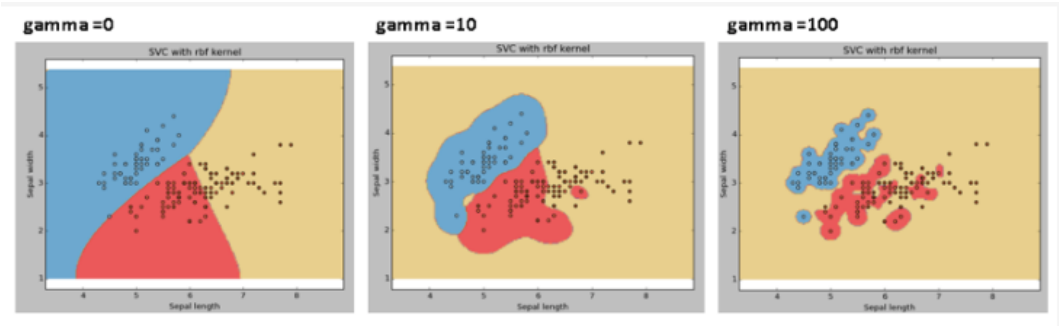
Gambar 3.2 Perbandingan Nilai C

Gamma secara teknis bukanlah parameter dari SVM. Gamma adalah parameter dari kernel yaitu pada kernel RBF (*Radial Basis Function*). Dengan menganggap kernel RBF untuk membangun sebuah *hypersphere* di sekitar setiap titik, yang kepadatannya menggunakan fungsi Gaussian. Gamma mengontrol fungsi standar deviasi dari Gaussian (bisa dilihat dari formula 3.11), formula tersebut nantinya akan dimasukkan dalam fungsi kernel (Vapnik, 1999).

Berikut adalah formula untuk parameter gamma (γ) dengan σ yaitu sigma yang merupakan sebuah *free parameter*:

$$\gamma = \frac{1}{2\sigma^2} \tag{3.11}$$

Semakin banyak nilai gamma maka semakin menyempit *hypersphere*-nya. Perhatikan perbandingan gamma pada gambar 3.3 berikut: (Vapnik, 1999)



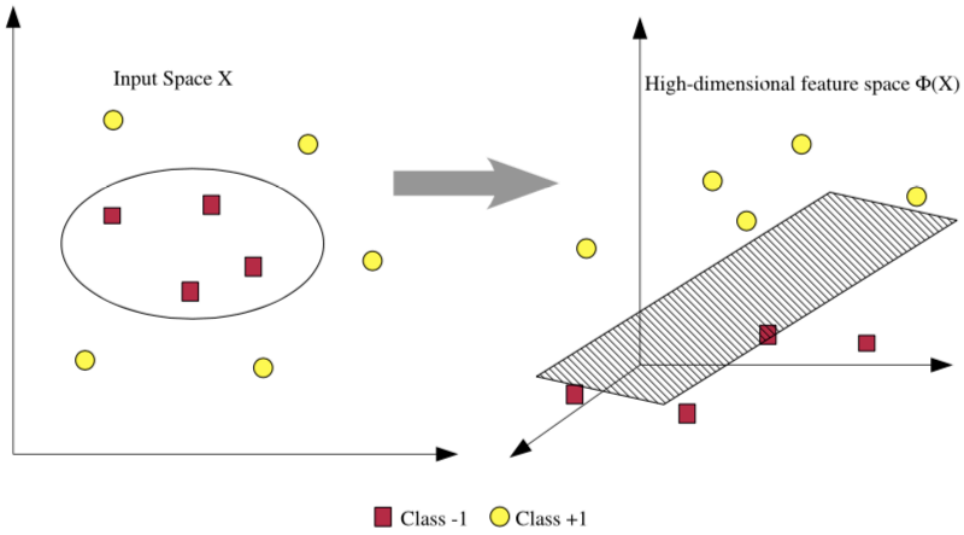
(Sumber : www.analyticsvidhya.com)

Gambar 3.3 Perbandingan Nilai Gamma

3.4. *Non Linear Support Vector Machine*

Sebagaimana dijelaskan pada bab sebelumnya, SVM merupakan satu varian dari *linear machine* sehingga hanya dapat dipakai untuk menyelesaikan masalah yang sifatnya *linearly separable*. Untuk dapat dipakai dalam kasus non-linear, pertama-tama data yang berada pada ruang vektor awal ($\{x_i \in R^d\}$) berdimensi d, harus dipetakan ke ruang vektor baru yang berdimensi lebih tinggi ($\{x'_i \in R^d\}$). Misalnya saja fungsi pemetaan tersebut dinotasikan sebagai $\Phi(x)$. Pemetaan ini bertujuan untuk merepresentasikan data ke dalam format yang *linear separable* pada ruang vektor yang baru. Ilustrasi proses ini ditunjukkan pada gambar 3.4. (Nugroho, 2007).

$$\Phi : R^d \rightarrow R^q \qquad d < q \tag{3.12}$$



(Sumber : Nugroho, 2007)

Gambar 3.4 Fungsi Φ memetakan data ke ruang vektor yang berdimensi lebih tinggi

Selanjutnya dilakukan proses *training* sama sebagaimana pada *Linear* SVM. Proses optimisasi pada fase ini memerlukan perhitungan *dot product* dua buah contoh diatas pada ruang vektor baru. *Dot product* kedua buah vektor (x_i) dan (x_j) dinotasikan sebagai $\Phi(x_i).\Phi(x_j)$. Nilai *dot product* kedua buah vektor ini dapat

dihitung secara tak langsung, yaitu tanpa mengetahui fungsi transformasi Φ. Teknik komputasi ini disebut *Kernel Trick*, yaitu menghitung *dot product* dua buah vector di ruang vektor baru dengan memakai komponen kedua buah vektor tersebut di ruang vektor asal sebagai berikut : (Nugroho, 2007)

$$K(x_i, x_j) = \Phi(x_i).\Phi(x_j) \tag{3.13}$$

Berbagai jenis fungsi dapat dipakai sebagai kernel K, sebagaimana tercantum pada Tabel 3.2.

Selanjutnya klasifikasi non linear pada SVM terhadap test sample *x* dirumuskan sebagai berikut: (Nugroho, 2007)

$$f(\Phi(x)) = w.\Phi(x) + b \tag{3.14}$$

Tabel 3.2Fungsi kernel yang sering dipakai dalam SVM

Nama *Kernel*	Definisi
Polynomial	$K(x_i, x_j) = (x_i.x_j + 1)^p$
Gaussian	$K(x_i, x_j) = exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$
Sigmoin	$K(x_i, x_j) = tanh(\alpha\, x_i.x_j + \beta)$

Support vector adalah subset dari data *training*, dengan α yang tidak negatif $\{x_i | \alpha_i \neq 0\}$ (i = 1,2,⋯,*l*).

Selain itu berikut adalah prosedur algoritma dari *Support Vector Machine*, yaitu sebagai berikut:

a. mengidentifikasi *hyperplane*.
b. menemukan *hyperplane* untuk memisahkan angggota antar kelas.
c. Menentukan data *training* dan data *testing*. data sampel diambil dari seluruh dataset sebanyak 66% secara random, kemudian data sampel ini

diidentifikasi sebagai data *training*, sedangkan sisanya yaitu 33% dijadikan data *testing*. Berikut adalah syntax R yaitu:

```
set.seed(1)

index = 1:nrow(dat)

sampel = sample(index,trunc(length(index)/3))

test = dat[sampel,]

train = dat[-sampel,]
```

d. Setelah itu memasukkan data kedalam fungsi SVM, dengan variabel data predikat sebagai variabel independen terhadap semua faktor rekam akademis sebagai variabel dependen dan dengan menentukan parameter C dan gamma, serta kernel RBF. Berikut adalah syntax R yaitu:

```
data.svm=
svm(trainSplit$data.Predikat~.,trainSplit,kernel=
"radial", cost= 10, gamma=100)
```

e. Pengujian data prediksi berdasarkan model SVM yang telah dibuat pada poin d, dan membuatnya menjadi tabel confusion matrix serta mengeluarkan output nilai akurasinya dan total data *testing*-nya, Berikut adalah syntax R yaitu :

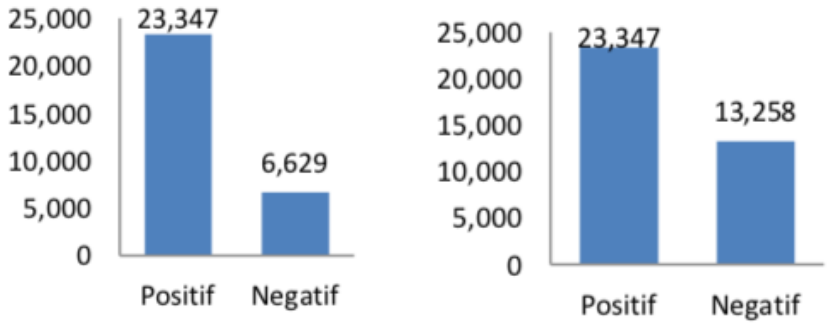
```
pred2 <- predict(data.svm,trainSplit)

cm2 <- table(pred2,trainSplit$data.Predikat)

str(trainSplit$data.Predikat)

accuracy2 <- sum(cm2[1,1]+cm2[2,2])/sum(cm2)

list(Matrix_Uji=cm2,Akurasi=accuracy2,Total_data=sum(cm2))
```

3.5. *Synthetic Minority Oversampling Technique (SMOTE)*

Synthetic Minority Oversampling Technique (SMOTE) adalah salah satu turunan dari oversampling. SMOTE pertama kali diperkenalkan oleh Nithes V. Chawla. Pendekatan ini bekerja dengan membuat replikasi dari data minoritas. Replikasi tersebut dikenal dengan data sintetis (*syntetic data*). Penerapan SMOTE yaitu dengan meminimalisasi keseimbangan kelas sehingga diharapkan dapat memiliki model yang baik (Siringoringo, 2018).

Kebaikan model klasifikasi dipengaruhi salah satunya oleh adanya keseimbangan antara kelas mayor dengan kelas minor. Kelas mayor adalah data yang ukuran kelasnya (jumlah amatan) lebih besar dari kelas minor berdasarkan peubah respon. Jika data yang digunakan untuk membuat model tidak seimbang maka akan meningkatkan salah klasifikasi kelas minor. Oleh karena itu, salah satu alternatif untuk meningkatkan akurasi model adalah melakukan *Synthetic Minority Oversampling Technique* (SMOTE) pada praposes (Barro, et al., 2013).

Pada contoh kasus Penerapan SMOTE meminimalisasi ketidak seimbangan kelas pada dataset *Credit Card Fraud* dengan membangkitkan data sintesis, sehingga total keseluruhan data terdiri dari 36.6056 data, yaitu 23.347 merupakan data positif dan 13.258 merupakan data negatif dapat dilihat pada gambar 3.5 dibawah setelah dilakukan SMOTE. Terlihat bahwa pada variabel negatif naik menjadi 13.258 data dari yang sebelumnya hanya 6.629 (Siringoringo, 2018).



(Sumber : Siringoringo, 2018)

Gambar 3.5 Kiri Grafik ketidak seimbangan awal dan Kanan Grafik ketidak seimbangan akhir dengan SMOTE

Metode SMOTE menambah jumlah data kelas minor agar setara dengan kelas mayor dengan cara membangkitkan data buatan. Data buatan atau sintesis tersebut dibuat berdasarkan k-tetangga terdekat (*k-nearest neighbor*). Jumlah k-tetangga terdekat ditentukan dengan mempertimbangkan kemudahan dalam melaksanakannya. Pembangkitan data buatan yang berskala numerik berbeda dengan kategorik. Data numerik diukur jarak kedekatannya dengan jarak euclidean sedangkan data kategorik lebih sederhana yaitu dengan nilai modus. Dengan persamaan ketetanggaan terdekat menggunakan perhitungan komputasi dengan memodifikasi rumus *dari Value Difference Metric* (VDM) atau nilai perbedaan metrik yang dirujuk oleh Cost dan Sailzberg (1993). Metrik berbeda dengan matrik, metrik merupakan alat untuk memperoleh pengukuran kuantitatif atau perkiraan pada kasus kualitatif (biasanya digunakan pada *software engineering*). VDM tersebut akan mengatasi *overlap* pada *feature value* pada *feature vectors*. Maka dibuatlah matrix untuk mendefinisikan jarak (δ) antara *feature value* pada *feature vectors*:

$$\delta(V_1, V_2) = \sum_{i=1}^{n} \left| \frac{C_{1i}}{C_1} - \frac{C_{2i}}{C_2} \right|^k \tag{3.15}$$

Dimana:

$\delta(V_1, V_2)$: jarak antar pengamatan (*feature values*) X dan Y pada peubah ke-*i*

C_{1i} : banyaknya kategori ke-1 yang termasuk kategorik peubah penjelas ke-*i*

C_{2i} : banyaknya kategori ke-2 yang termasuk kategorik peubah penjelas ke-*i*

C_1 : banyaknya kategori ke-1

C_2 : banyaknya kategori ke-2

n : banyaknya kategori pada peubah penjelas ke-i

k : konstanta (Biasanya diset 1)

Pada persamaan 3.15 digunakan untuk meghitung matriks perbedaan nilai pda setiap pengamatan dalam set vector pengamatan yang diberikan. Persamaan 3.15 digunakan pada jarak geometris yang telah ditetapkan dan memiliki himpunan nilai yang terbatas. Kemudian untuk mengatasi hal tersebut maka Cost dan Sailzberg

(1993) memodifikasi *Value Difference Metric* (VDM) dengan memberikan bobot (*w*) pada persamaan jarak (δ) dimana nantinya akan membuat jarak menjadi simetris. Jarak (Δ) antara dua vektor pengamatan yaitu:

$$\Delta(X,Y) = w_x w_y \sum_{i=1}^{N} \delta(x_i, y_i)^r \quad (3.16)$$

Dimana:

$\Delta(X,Y)$: Jarak amatan X dan Y

w_x w_y : bobot pada modifikasi VDM

N : banyaknya variabel prediktor

r : bernilai 1 jika menggunakan jarak *Manhattan* dan r = 2 jika jarak menggunakan *Euclidean*.

Selain itu juga berikut adalah algoritma prosedur dengan menggunakan SMOTE. Algoritma SMOTE digunakan sebelum preprosesing sebelum melakukan permodelan fungsi SVM, algoritma SMOTE ini akan me-*resampling* dari data training sehingga nantinya jumlah data training setelah dilakukan SMOTE akan berkurang sesuai dengan berapa persen dilakukan *sampling*. SMOTE dilakukan pada variabel predikat sebagai variabel independent terhadap faktor rekam akademis sebagai variabel dependen, dengan menentukan jumlah presentase untuk *oversampling* pada kelas minor 800 artinya dari data *training* kelas minor akan di *over resampling* sebanyak 8% dari total data *training* dan *undersampling* pada kelas mayor 1000 dari data *training* kelas mayor akan di *under resampling* sebanyak 10% dari total data *training* Berikut adalah syntax R yaitu :

```
trainSplit <- SMOTE(data.Predikat ~ ., train, perc.over =
800, perc.under=1000)
```

Setelah data training di- *resampling* maka selanjutnya maka dilanjutkan dengan prosedur SVM seperti pada sub bab 3.4.

3.6. *Confusion Matrix*

Pengukuran terhadap kinerja suatu sistem klasifikasi merupakan hal yang penting. Kinerja sistem klasifikasi menggambarkan seberapa baik sistem dalam mengklasifikasikan data. *Confusion matrix* merupakan salah satu metode yang dapat digunakan untuk mengukur kinerja suatu metode klasifikasi. (Prasetyo, 2012)

Pada *Confusion matrix* akan melakukan beberapa rumus ini melakukan perhitungan dengan 4 keluaran, yaitu: *recall, precision, acuraccy* dan *error rate*. (Prasetyo, 2012)

Tabel 3.3 Bagian *Confusion Matrix*

		Predicted	
		Negative	*Positive*
Actual	*Negative*	TP	FN
	Positive	FP	TN

- *Recall* adalah proporsi kasus positif yang diidentifikasi dengan benar. Dengan rumus:

$$Recall = \frac{TP}{(FN+TP)} \tag{3.17}$$

Dan untuk *multi-class* berikut adalah rumus *recall* :

$$Recall\ multi-class = \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l} (TP_i+FN_i)} \tag{3.18}$$

- *Precision* adalah proporsi kasus dengan hasil positif yang benar. Dengan rumus:

$$Presicion = \frac{TP}{(FP+TP)} \tag{3.19}$$

Dan untuk *multi-class* berikut adalah rumus *presicion*nya :

$$Presicion\ multi-class = \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l} (FP_i+TP_i)} \tag{3.20}$$

- *Accuracy* adalah perbandingan kasus yang diidentifikasi benar dengan jumlah semua kasus. Dengan rumus:

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (3.21)$$

Dan untuk *multi-class* berikut adalah rumus *recall* :

$$Accuracy\ multi-class = \sum_{i=1}^{l} \frac{TP_i+TN_i}{TP_i+TN_i+FP_i+FN_i} \quad (3.22)$$

- *Error Rate* adalah kasus yang diidentifikasi salah dengan sejumlah semua kasus. Dengan rumus:

$$Error\ Rate = \frac{(FN+FP)}{(TP+TN+FP+FN)} \quad (3.23)$$

Dan untuk *multi-class* berikut adalah rumus *recall* :

$$Error\ multi-class = \sum_{i=1}^{l} \frac{FN_i+FP_i}{TP_i+TN_i+FP_i+FN_i} \quad (3.24)$$

Keterangan:

- TP adalah *True Positive*, yaitu jumlah data positif yang terklasifikasi dengan benar oleh sistem.
- TN adalah *True Negative*, yaitu jumlah data negatif yang terklasifikasi dengan benar oleh sistem.
- FN adalah *False Negative*, yaitu jumlah data negatif namun terklasifikasi salah oleh sistem.
- FP adalah *False Positive*, yaitu jumlah data positif namun terklasifikasi salah oleh sistem.
- TP_i adalah *True Positive*, yaitu jumlah data positif yang terklasifikasi dengan benar oleh sistem untuk kelas ke-*i*.
- TN_i adalah *True Negative*, yaitu jumlah data negatif yang terklasifikasi dengan benar oleh sistem untuk kelas ke-*i*.
- FN_i adalah *False Negative*, yaitu jumlah data negatif namun terklasifikasi salah oleh sistem untuk kelas ke-*i*.
- FP_i adalah *False Positive*, yaitu jumlah data positif namun terklasifikasi salah oleh sistem untuk kelas ke-*i*.
- *l* adalah jumlah kelas.