BAB II

LANDASAN TEORI

2.1 Data Mining

Data mining merupakan rangkaian proses penting untuk memperoleh informasi atau pengetahuan yang dari suatu kumpulan dataset yang kompleks. *Data mining* memiliki tujuan utama yaitu menemukan pola, keterhubungan, atau informasi yang mungkin tidak tampak secara langsung dalam data, sehingga mendapatkan manfaat dari wawasan yang telah diberikan secara mendalam.

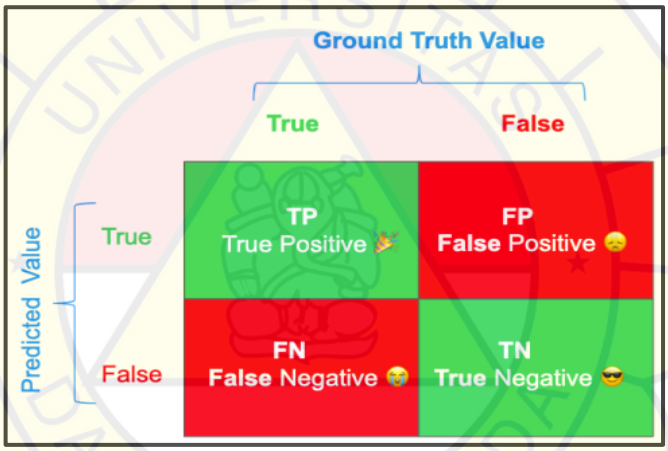
Pada proses *data mining* mencakup penerapan dari berbagai teknik statistik, matematis, dan kecerdasan buatan dengan tujuan untuk menganalisis data secara sistematik dan otomatis. Hasil yang diperoleh dari *data mining* dapat digunakan sebagai dukungan pengambilan keputusan, mengenali tren pasar, meningkatkan efisiensi operasional serta merumuskan startegi bisnis.

2.2 Teknik-Teknik Data Mining

Berikut beberapa teknik umum dalam *data mining* yang digunakan untuk mengekstrak pola dan pengetahuan dari data:

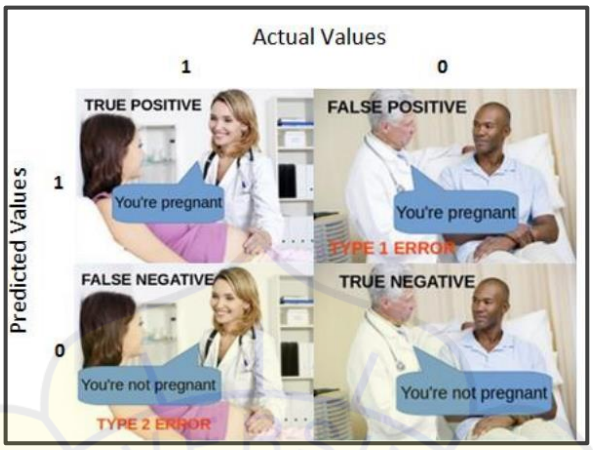
1. Klasifikasi (*Classification*), salah satu metode yang dengan menggunakan teknik pengelompokan data ke satu kelas atau kategori khusus dari atribut atau karakter tertentu. Tujuan dalam metode ini adalah untuk menemukan pola atau keterkaitan yang terdapat di dalam data, sehingga dapat membuat keputusan atau prediksi. Teknik klasifikasi adalah tahap pembelajaran dari suatu fungsi tujuan (target) dengan menghubungkan atribut x kesalah satu label kelas y yang sudah ditetapkan. Teknik yang tepat untuk berbagai data biner atau

nominal. Contoh yang terdapat dalam metode ini mencakup pohon keputusan atau *decision tree, Naïve Bayes, Support Vector Machine,* serta algoritma klasifikasi lainnya. Untuk menilai kinerja algoritma klasifikasi, pentingnya untuk memakai *Confusion Matrix* sebagai penguji performa sebagai pemantau nilai prediksi serta nilai terbaru dalam *model supervised* learning klasifikasi di bidang *data mining*. Dari langkah klasifikasi ini mengeluarkan empat pembanding antara nilai prediksi dan nilai terbaru, seperti *True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN)*. Berikut perbandingan *Confusion Matrix*:



Gambar 2.1 Confusion Matrix

Berdasarkan **Gambar 2.1**, *TP* dan *TN* merupakan klasifikasi yang diinginkan namun *FP* dan *FN* ialah kesalah yang bisa saja terjadi. Di bawah ini terdapat **Gambar 2.2** yang merupakan ilustrasinya:



Gambar 2.2 Ilustrasi Confusion Matrix

Berikut uraian dari **Gambar 2.2** ilustrasi *Confusion Matrix*:

a. *True Positive*: Dapat memperkirakan hal positif dan sesuai dengan kenyataan. Seperti memperkirakan seorang wanita sedang hamil dan hal itu sesuai dengan kenyataannya.
b. *True Negative*: Dapat memperhitungkan hal negatif dan sesuai dengan kenyataan. Contohnya memperkirakan seorang pria hamil namun pada kenyataannya seorang pria tidak bisa hamil.
c. *False Positive*: Merupakan tipe kesalahan 1, yang menjelaskan bahwa dapat memperkirakan hal positif namun bertolak belakang dengan kenyataan, dimana pada kenyataan menyatakan negative.
d. *False Negative*: Merupakan tipe kesalahan 2, dimana memperkirakan hal negatif dan bertolak belakang dengan kenyataan, yang artinya di kenyataan menyatakan positif.

Akurasi, presisi serta *recall* merupakan tiga hal matrix yang sering dipakai untuk penilaian di suatu model klasifikasi, untuk menguji seberapa efisien kinerja model klasifikasi tersebut. Berikut penjelasan dari ketiga matrix tersebut:

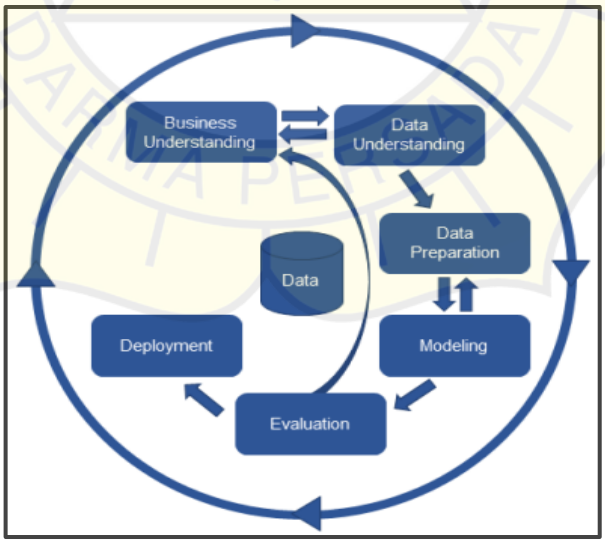
a. Akurasi, merupakan *matriks* yang digunakan untuk pengujian seberapa akurat model klasifikasi dalam memprediksi kelas dengan baik. Contoh rumusnya adalah: (Jumlah prediksi yang benar) / (Jumlah total prediksi)
b. Presisi, ialah *matrix* yang digunakan untuk pengujian seberapa jauh dalam memprediksi positif berdasarkan model yang ditetapkan sudah tepat dan akurat. Berikut adalah rumusnya: (Jumlah prediksi positif yang akurat) / (Jumlah total prediksi positif)
c. *Recall*, ialah matrix yang menguji sejauh mana model mampu mengidentifikasi semua data masuk ke dalam kelas yang sudah ditetapkan. Rumusnya sebagai berikut: (Jumlah prediksi positif yang akurat) / (Jumlah total contoh yang sudah ada dalam kelas yang ditentukan).

2. Regresi (*Regression*), teknik yang digunakan sebagai pemodelan hubungan antara variabel dependen dan independent. Teknik ini menentukan prediksi nilai kontinu berdasarkan hubungan linier atau non-linear antara variabel tersebut berdasarkan data histori.
3. Klastering (*Clustering*), teknik satu digunakan untuk pengelompokan data ke dalam klister ataupun kelompok menurut kesamaan atribut atau fitur yang sudah ditetapkan Contoh yang terdapat dalam teknik mencakup Algoritma *K-Means* dan *Clustering Hierarchical*.
4. Asosiasi (*Association*), dalam teknik ini digunakan untuk menemukan hubungan atau asosiasi antara variabel atau item di dalam dataset. Asosiasi berguna dalam

menemukan pola keterkaitan yang dapat memberikan wawasan lebih detail berdasarkan dataset. Dengan menggunakan algoritma *Apriori* dan *Eclat* yang digunakan untuk aturan asosiasi dalam data transaksi. Metode ini diterapkan dalam berbagai bidang seperti ritel, manajemen pasokan, pemasaran, keamanan jaringan, kesehatan, analisis panen, edukasi dan lain sebagainya. (Rahayu, P. W., Sudipa, I. G. I., Suryani, A., Surachman, A., Ridwan, A., Darmawiguna, I. G. M., Sutoyo, M. N., Slamet, I., Harlina, S., & Sanjaya, 2024)

2.3 CRISP-DM

CRISP-DM atau *Cross-Industry Standard Process for Data Mining*, dikembangkan pada tahun 1996 oleh analis dari berbagai industry seperti *NCR, Daimler Chrysler,* dan *SPSS,* yang mencakup kerangka kerja dari data mining dengan tujuan memecahkan masalah dalam berbagai kondisi bisnis ataupun untuk keperluan penelitian. (Pradnyana, G. A., & Agustini, 2022)



Gambar 2.3 Tingkatan proses CRISP-DM

Kerangka kerja berdasarkan **Gambar 2.3** tersebut memiliki enam tingkatan proses yang saling berurutan dan bersifat adaptif, selain itu kerangka kerja *CRISP-DM* bergantung dengan keluaran proses sebelumnya. Berikut uraian dari keenam tingkatan proses *CRISP-DM:*

1. **Proses Pemahaman Bisnis atau Business Understanding Process:**

   Tahap ini memahami proses penelitian dengan menyusun dan mempersiapkan strategi untuk mencapai tujuan penelitian atau bisnis. Dalam penelitian ini, peneliti mempersiapkan data-data serta penilaian dalam menentukan mahasiswa yang berpotensi *Drop Out* di program studi teknologi informasi Universitas Darma Persada. Mahasiswa *Drop Out* bisa dilihat dari kategorinya seperti masa kuliah lebih dari 7.5 tahun serta Indeks Prestasi Kumulatif (IPK) < 2.0. Studi kasus yang dilakukan pada penelitian menggunakan fitur persentase kehadiran, riwayat tagihan mahasiswa semester 1 hingga semester 4 serta Indeks Prestasi Semester (IPS) 1 hingga 4.

2. **Proses Pemahaman atau Data Understanding Process:**

   Pemahaman data dengan cara mengumpulkan data mahasiswa angkatan 2018 melalui akses portal akademik kemudian dianalisis dengan tujuan mendapatkan pemahaman lebih lanjut serta evaluasi kualitas data. Penelitian ini diuji coba berdasarkan fitur yang ditetapkan yaitu persentase kehadiran, riwayat tagihan mahasiswa semester 1 hingga semester 4 serta Indeks Prestasi Semester (IPS) 1 hingga 4.

3. **Proses Persiapan Data atau Data Preparation Process:**

   Data awal disiapkan dengan tujuan untuk proses selanjutnya dengan menentukan variabel yang sesuai kemudian dianalisis apakah variabel tersebut

perlu diubah atau tidak. Tahap ini juga dapat disebut dengan data pra pemrosesan, *Data preprocessing* atau disebut juga sebagai pra-pemrosesan data merujuk pada deretan proses yang dilakukan pada data mentah sebelum digunakan untuk analisis lebih lanjut atau untuk pembuatan model. Tujuannya yaitu untuk meningkatkan kualitas data, menjamin keakuratan hasil analisis serta mengatasi masalah atau kekurangan yang mungkin muncul dalam data mentah. Adapun bagian utama dari pra-pemrosesan data, sebagai berikut:

1. *Data Cleaning* (Pembersihan Data), suatu langkah identifikasi dan mendeteksi serta mengatasi data yang tidak sesuai yang dapat mempengaruhi hasil analisis. Proses ini juga dapat menangani duplikasi dan menghapus data duplikat yang menyebabkan hasil yang tidak konsisten dan tidak akurat. Dalam penelitian ini, *data cleaning* yang dilakukan dengan mengatasi jumlah data yang tidak seimbang antara data 'Lulus' yang berjumlah 208 dan *'Drop Out'* yang berjumlah 108. Masalah data yang tidak seimbang ini dilakukan dengan menggunakan SMOTE atau Synthetic Minority Oversampling Technique, dengan cara menemukan jumlah data 'Lulus' dan 'Drop Out', kemudian mengimlementasi teknik SMOTE agar data menjadi seimbang, dan melakukan pemeriksaan ulang pada data setelah SMOTE diterapkan. Dengan menerapkan teknik SMOTE jumlah data *'Drop Out'* dilakukan peningkatan agar memiliki jumalh yang sama dengan jumlah data 'Lulus' Maka, jumlah data 'Lulus' ataupun *'Drop Out'* memiliki jumlah yang sama yaitu sebanyak 208 data.

2. *Missing Values Handling* (Pengisian Nilai yang hilang), proses ini dengan menggunakan metode tertentu seperti nilai rata-rata, hal ini dilakukan untuk mengisi atau memperkirakan nilai yang hilang.
3. *Data Integration* (Integrasi Data), menggabungkan data dari berbagai sumber dengan tujuan menghasilkan kumpulan data yang lebih lengkap dan relevan. Selain itu, mengatasi format, skema atau struktur data yang berbeda.
4. *Data Transformation* (Transformasi Data), mengubah bentuk distribusi dan skala data agar lebih konsisten, dan mengubah variabel kategorikal menjadi bentuk yang dapat diolah oleh algoritma.
5. *Feature Selection* (Seleksi Fitur), memilih bagian fitur yang paling penting dan relevan untuk pemodelan atau analisis.
6. *Handling Imbalanced Data* (Manajemen Data Tidak Seimbang), meningkatkan atau mengurangi jumlah sampel dalam kelas tertentu yang berutjuan mengatasi masalah ketidakseimbangan kelas. Selain itu, memperbaiki ketidaksesuaian data yang dapat mempengaruhi akurasi analisis.
7. Optimasi Kinerja Pemrosesan Paralel, teknik ini diterapkan untuk meningkatkan kecepatan dan efisiensi dalam pemrosesan data.

4. Proses Modeling atau Modelling Process:

Menerapkan teknik pemodelan *data mining* yang tepat, jika perlu proses dapat kembali ke tahap sebelumnya yaitu proses persiapan data. Hal itu bertujuan untuk membentuk teknik *data mining* yang lebih tepat yang akan digunakan. Dalam Penelitian ini model yang digunakan adalah metode klasifikasi dengan

algoritma *Support Vector Machine* dan *Decision Tree.* Dari kedua algoritma klasifikasi tersebut, menunujkan bahwa metode *SVM* memiliki akurasi yang lebih tinggi dari metode *Decision Tree.* Algoritma tersebut didapatkan dari hasil uji coba model yang sesuai dengan data yang ingin diprediksi atau di klasifikasi dengan menggunakan *Jupyter Notebook.*

5. **Proses Evaluasi atau Evaluation Process:**

   Melakukan penilaian dalam beberapa model yang dipakai pada proses sebelumnya sebagai uji coba kualitas serta efektivitas seperti menetapkan model serta tujuan sebelum melakukan proses selanjutnya. Penilaian atau evaluasi ini dapat dilakukan dengan memakai data latih dan data uji yang dipisahkan dari data asli dengan perbandingan 80:20. Dalam tahap penilaian dan evaluasi menghasilkan *confusion matrix*, presisi, serta akurasi untuk menilai kinerja model.

6. **Proses penyebaran atau Deployment Process:**

   Model yang dihasilkan dimanfaatkan dalam konteks yang berkaitan, sebagai contoh proses penyebaran *data mining* dan memantau performa berdasarkan proses tersebut. (Pradnyana, G. A., & Agustini, 2022) Dalam tahap ini menerapkan metode tersebut ke dalam aplikasi yang ingin dikembangkan.

2.4 Metode Decision Tree

Salah satu teknik metode yang sering dipakai adalah metode *Decision Tree*. *Decision tree* terstruktur seperti pohon dimana dalam simpul internal menunjukan uji coba di suatu atribut, cabang-cabangnya menunjukan hasil uji coba, dan simpul daun yang menghasilkan kelas atau distribusi kelas. Untuk memprediksi kelas data dibuat dengan mengikuti alur berdasarkan simpul akar ke simpul daun. Selain itu,

Decision Tree dengan mudah diubah menjadi model klasifikasi yang lebih mudah dipahami.

Dalam penerapan, membagi serta mengontrol strategi. Teknik metode *decision tree* ini efisien dalam parameter yang kemudian digunakan untuk klasifikasi serta regresi dalam membangun pohon dengan label berdasarkan data pelatihan. Selain itu, digunakan untuk konversi menjadi sederet aturan yang mudah untuk dipahami. *Decision tree* merupakan salah satu teknik metode yang akurat serta populer dalam penerapan *data mining* yang digunakan untuk klasifikasi dan memprediksi. Dalam uraian tersebut dapat dimengerti bahwa pada metode *decision tree*, data dapat diubah menjadi sebuah keputusan serta aturan keputusan.

Terdapat berbagai manfaat yang diberikan dari metode *decision tree* atau pohon keputusan, seperti berguna dalam melakukan berbagai tugas *data mining* seperti klasifikasi, regresi, *clustering* dan seleksi fitur pada data tertentu. Metode pohon keputusan ini tergolong mudah diikuti serta memiliki fleksibilitas dalam mengatasi berbagai jenis input data seperti data berjenis nominal, numerik beserta kategori, penyesuaian dalam mengatasi dataset yang mungkin terdapat nilai yang memiliki kesalahan atau tidak lengkap, memiliki kemampuan prediksi yang akurat. Terdapat berbagai paket *data mining* yang dapat diakses dari berbagai platform dan bermanfaat untuk dataset yang kompleks terutama konteks kerangka *ensemble*.

Metode ini juga mempunyai peran penting dalam mengeksplorasi data dalam mengidentifikasi hubungan yang tidak terlihat antara beberapa variabel *input* dan variabel target. Terdapat beberapa algoritma yang dapat dimanfaatkan

dalam membentuk pohon keputusan seperti *ID3, CART* dan *C4.5*. Berikut contoh dataset yang terdiri dari banyak data dengan tujuan untuk menilai apakah seorang mahasiswa tersebut lulus dengan waktu atau tidak. (Rahayu, P. W., Sudipa, I. G. I., Suryani, A., Surachman, A., Ridwan, A., Darmawiguna, I. G. M., Sutoyo, M. N., Slamet, I., Harlina, S., & Sanjaya, 2024)

Tabel 2.1 Dataset Kelulusan Mahasiswa

NIM	Gender	Nilai UN	Asal Sekolah	IPS1	IPS2	IPS3	IPS 4	Ket. Lulus
10001	L	28	SMAN 2	3.3	3.6	2.89	2.9	Ya
10002	P	27	SMAN DK	4.0	3.2	3.8	3.7	Tidak
10003	P	24	SMAN 1	2.7	3.4	4.0	3.5	Tidak
…	…							
11000	L	23.4	SMAN 5	3.3	2.8	3.1	3.2	Ya

Dalam **table 2.1** menggunakan algoritma *Decision Tree C4.5* merupakan hal yang sudah tepat. Dengan menggunakan algoritma tersebut akan menghasilkan berupa pohon keputusan yang dapat memberi pemahaman detail terkait penyebab atau faktor yang mempengaruhi kelulusan mahasiswa adalah tepat waktu. Dalam membaca hasil dari perhitungan algoritma *C4.5*, diketahui bahwa status mahasiswa yang menentukan awal apakah mahasiswa dapat lulus tepat waktu. Misal, mahasiswa berstatus bekerja, maka langkah selanjutnya adalah Indeks Prestasi Semester (IPS) 2. Jika nilai IPS 2 lebih tinggi dari 3.5 maka akan ditentukan prediksi sebagai penilaian. Namun, jika IPS 8 lebih tinggi dari 2.9 maka yang akan diperiksa tahap selanjutnya adalah IPS 3. Jika IPS3 bernilai kecil dan kurang dari 2.57, maka akan lulus tepat waktu, tetapi jika IPS 3 lebih tinggi

nilainya dari 2.57 maka diprediksi mahasiswa akan berpotensi lulus terlambat. Keputusan dari *decision tree* sangat berkaitan dengan data yang digunakan. Semakin banyak jumlah datanya maka semakin lengkap dan akurat pula hasil dari *decision tree*. Sebaliknya jika datanya sedikit maka akan menghasilkan akurasi yang tidak maksimal.

2.4.1 Algoritma C4.5

Algoritma *C4*.5 diperkenalkan oleh Quinlan, sebagai model terbaru yang memperbaiki dari *ID3*. Dalam *ID3*, *decision tree* dibuat hanya untuk fitur dengan tipe data kategorikal (nominal/ordinal), tetapi untuk fitur tipe numerik (internal/rasio) tak bisa dipakai. Hal yang menjadi pembeda antara *C4.5* dan *ID3* ialah dapat mengatasi fitur yang bertipe data numerik, memotong (*pruning*) *decision tree*, serta mengeluarkan aturan atau *deriving rule set*. Dalam algoritma ini juga memakai kriteria gain untuk menetapkan fitur yang menjadi pemisah node dalam pohon keputusan yang telah dibuat. (Muslim, M. A., Prasetiyo, B., Mawarni, E. L. H., Herowati, A. J., Mirqotussa'adah, Rukmana, S. H., & Nurzahputra, n.d.)

2.4.2 Entropy

Entropi dimanfaatkan untuk menetapkan *node* mana yang akan dipakai untuk pemisah pada data pelatihan selanjutnya. Nilai entropi yang besar dapat meningkatkan akurasi klasifikasi. Perlu diawasi bahwa entropi bernilai 0 maka menunjukkan semua *vector* data terletak pada satu label kelas serta *node* tersebut membentuk simpul daun yang berisikan keputusan atau label kelas. Dalam perhitungan entropi adalah jika salah satu dari elemen terdapat jumlah

0, maka entropi tersebut juga bernilai 0. Lalu, jika proporsi dari semua elemen berjumlah sama maka akan ditetapkan nilai entropi menjadi 1 (Muslim, M. A., Prasetiyo, B., Mawarni, E. L. H., Herowati, A. J., Mirqotussa'adah, Rukmana, S. H., & Nurzahputra, n.d.). Selain itu, metode perhitungan nilai entropi akan ditetapkan dalam persamaan dibawah ini:

$$Entropi(S) = \sum_{i=1}^{n} -p_i * \log_2 p_i$$

Dengan keterangan berikut:

S = Himpunan Kasus

n = jumlah partisi S

p_i= Proporsi dari S_i terhadap S

Dimana $\log_2 p_i$ perthitungannya akan dilakukan dengan metode yang sama dengan persamaan dibawah ini:

$$log(X) = \frac{ln(x)}{ln(2)}$$

2.4.3 Gain Ratio

Menurut (Prasetyo, 2014: 67), menyatakan bahwa, salah satu kriteria yang sering dipakai dalam pemilihan fitur yang digunakan untuk pemisah dalam algoritma *C4.5* berdasarkan gain ratio dapat dijelaskan dengan persamaan dibawah ini:

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}$$

Dalam perhitungan gain dapat menggunakan persamaan berikut ini:

$$Gain(S, A) = Entropy(S) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} * Entropy(S_i))$$

Dengan keterangan:

S = Himpunan Kasus

A = Atribut

n = jumlah partisi atribut A

$|S_i|$ = jumlah kasus pada partisi ke-i

$|S|$ = jumlah kasus dalam S

Namun jika ingin menghitung *Split Entropy* menggunakan persamaan dibawah ini:

$$Split_{info}(S) = -\sum_{i=1}^{n} \frac{|S_i|}{|S|} * \log_2 \frac{|S_i|}{|S|}$$

Dengan keterangan:

S = Himpunan Kasus

A = Atribut

n = jumlah partisi atribut A

$|S_i|$ = jumlah kasus pada partisi ke-i

$|S|$ = jumlah kasus dalam S

2.4.4 Contoh Studi Kasus

Berikut ini terdapat contoh studi kasus untuk menghitung dengan memakai algoritma *C4.5* berdasarkan dataset diabetes mellitus yang didapat dari *UCI Machine Learning Repository*. Sebelum menjalankan klasifikasi, harus melakukan proses pengolahan data yang mencakup *data cleaning* dan *data transformation*, lalu mengeluarkan data baru yang siap untuk ditindaklanjuti dengan menggunakan algoritma *C4.5*. Keunggulan dari algoritma *C4.5*, yaitu mampu dalam mengatasi fitur dengan tipe numerik. Tahap pertama setelah data

diolah yaitu memisahkan data ke bentuk *data training* bertujuan untuk membentuk pohon keputusan serta *data testing* untuk menguji kinerja pohon tersebut. Pemisahan data tersebut memakai *k-fold cross validation* dengan nilai bawaan yaitu nilai k = 10.

Berikut merupakan contoh dalam menganalisis data dengan tahap-tahap membentuk pohon keputusan dengan algoritma *C4.5*:

a. Mempersiapkan *data training,* yang biasanya didapatkan dari data histori atau data sebelumnya yang sudah terjadi sebelumnya serta telah digabungkan dengan kelas khusus.
b. Perhitungan jumlah kasus, termasuk jumlah kasus untuk keputusan *tested_positve* dan jumlah kasus untuk *tested_negative* berdasarkan dari masing-masing atribut yang ditunjukkan pada **Tabel 2.2** di bawah ini.

Tabel 2.2 Jumlah Kasus dari masing-masing atribut

No	**Atribut**	**Nilai Atribut**	**Jumlah Kasus Total**	**Tested_Positive**	**Tested_Negative**
1	Pregnant	≤ *6* > 6	599 169	173 95	426 74
2	Plasma	≤ 127 > 127	485 283	94 174	391 109
3	Pressure	≤ 68 > 68	283 485	70 198	213 287
4	Skin	≤ 23 > 23	172 596	27 241	145 355
5	Insulin	≤ *87* > 87	118 650	9 259	109 391
6	BMI	≤ 27,3	183	18	165

		> 27,3	585	259	335
7	Pedigree	≤ *0,527* > 0,527	509 259	148 120	361 139
8	Age	≤ *28* > 28	367 401	71 197	296 204

c. Perhitungan nilai entropy, information gain, *split info* dan *gain ratio* secara keseluruhan kasus serta kasus dipisah berdasarkan tiap-tiap atribut. Selanjutnya, melakukan perhitungan nilai gain ratio untuk setiap atribut. Nilai *gain ratio* yang paling besar akan membentuk akar pertama. Cara perhitungannya akan dijelaskan pada table 2.3 di berikut ini.

Di bawah ini merupakan **tabel 2.3** yang berisikan hasil perhitungan nilai *gain ratio*.

Tabel 2.3 Hasil menghitung gain

No	Atribut	Jumlah kasus total	tested_positive	tested_negative	entropy	info gain	split info	gain ratio
1	class	768	268	500	0,9331			
2	Pregnant ≤ 6 > 6	 599 169	 173 95	 426 74	 0,8671 0,9888	0,0391	0,7602	0,0515
3	Plasma ≤127 >127	 485 283	 94 174	 391 109	 0,7093 0,9616	0,1308	0,9495	0,1377
4	Pressure ≤68 >68	 283 485	 70 198	 213 287	 0,807 0,9755	0,0196	0,9495	0,0207
5	Skin ≤23 >23	 172 569	 27 241	 145 355	 0,627 0,9734	0,0372	0,7673	0,0485
6	Insulin ≤87 >87	 118 650	 9 259	 109 391	 0,3889 0,97	0,0523	0,6188	0,0846
7	BMI					0,0725	0,7921	0,0915

	≤27,3 >27,3	183 585	18 250	165 335	0,4637 0,9847			
8	Pedigree ≤0,527 >0,527	 509 259	 148 120	 361 139	 0,8697 0,9961	0,0207	0,9221	0,0225
9	Age ≤28 >28	 367 401	 71 197	 296 204	 0,7086 0,9997	0,0724	0,9985	0,0725

d. Langkah untuk menghasilkan nilai entropy pada tiap atribut, pada tahap awal menghitung nilai entropy pada data secara menyeluruh untuk kelas (class = ”tested_positive” dan class = ”tested_negative”).

Total class target (class) = 2 (“tested_positive” dan “tested_negative”)

Total kasus tested_positive = 268

Total kasus tested_negative = 500

Jumlah total = 768

Berdasarkan hitungan yang serupa dapat diterapkan pada masing-masing atribut dengan menggabungkan total kasus di setiap atribut serta subset atribut di dalamnya. Untuk menghitung entropy dapat diperoleh berikut:

$$Entropy\ (Class)\ =\ 1(268{,}500)\ =\ -((\frac{268}{768})\log_2(\frac{268}{768}))-((\frac{500}{768})\log_2(\frac{500}{768}))$$

$$=0{,}933134317$$

e. Setelah itu, menghitung nilai gain pada setiap atribut. Nilai gain digunakan untuk atribut “pregnant”, perhitungannya dapat dilakukan sebagai berikut:

$$Gain(Class, Pregnant\ =\ Entropy\ (Class)\ -\ \sum_{i=1}^{n}\frac{|Jumlah|}{|Total|}\ *\ Entropy(Pregnant)$$

$$Gain(Class, Pregnant)\ =\ 0{,}9331\ -\ ((\frac{599}{768})*0{,}8671)-((\frac{169}{768})*0{,}9888)$$

$$Gain(Class, Pregnant)=0{,}039180261$$

f. Sedangkan untuk menghitung gain ratio membutuhkan nilai split info dengan cara berikut:

$$Split_{info}(Pregnant)\ =\ -\sum_{i=1}^{n}\frac{|Jumlah|}{|Total|}*\log_2\frac{|Jumlah|}{|Total|}$$

$$Split_{info}(Pregnant) = \left(\left(\frac{599}{768}\right) * \log_2\left(\frac{599}{768}\right)\right) + \left(\left(\frac{599}{768}\right) * \log_2\left(\frac{599}{768}\right)\right)$$

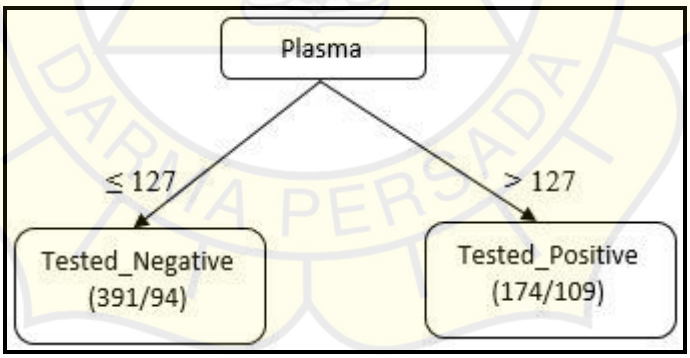
$$Split_{info}(Pregnant) = 0{,}760262594$$

g. Dan untuk menghitung gain ratio menggunakan persamaan berikut:

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}$$

$$GainRatio(A) = \frac{0{,}039180261}{0{,}760262594} = 0{,}051535169$$

Dalam **Gambar 2.4** dapat dilihat bahwa atribut tertinggi dihasilkan oleh gain ratio bagian plasma sebesar 0,1377. Maka dari itu plasma dapat dijadikan node akar. Untuk gambar pohon keputusannya dapat diilustrasikan sebagai berikut.



Gambar 2.4 Pohon keputusan hasil perhitungan node 1

Melakukan perhitungan total kasus, total kasus untuk keputusan *tested_positive*, jumlah kasus untuk keputusan *tested_negative*, entropi, *information gain*, *split info*, serta gain ratio telah dipilih. Kemudian, menghitung nilai gain ratio untuk setiap atribut. Setelah semua atribut

masuk ke dalam pohon dan memiliki kelas maka perhitungan lebih lanjut tidak diperlukan.

Berikut adalah studi kasus menggunakan *decision tree* pada penelitian ini. Berdasarkan dataset pada penelitian ini, terdapat beberapa atribut seperti IPS semester 1-4, riwayat tagihan semester 1-4 dan kehadiran mahasiswa. Pada studi kasus ini akan menerapkan atribut dengan *Gain Ratio* tertinggi dalam tiap langkah untuk membuat pohon keputusan.

Tabel 2.4 Studi Kasus Penelitian

No.	IPS				Tagihan				Kehadiran	Status Kelulusan
	1	2	3	4	1	2	3	4		
1.	2.35	2.50	2.25	2	0	0	0	0	90%	Lulus
2.	1.8	2.9	2.7	3.8	0	1	0	0	85%	Lulus
3.	3.2	2.11	2.90	3.25	0	0	0	0	75%	Lulus
4.	1.5	1.80	2	2.5	0	0	1	0	80%	DO
5.	3.8	3.7	3.9	3.8	0	0	0	0	88%	Lulus
6.	2.7	2.8	2.6	2.7	0	0	1	0	78%	Lulus
7.	2.6	2	1.85	2	0	1	1	1	65%	DO
8.	3.4	3.5	3.3	3.4	0	0	0	0	100%	Lulus
9.	1.66	2.9	3.0	3.0	0	0	0	0	75%	Lulus
10.	1.99	1.63	1.88	2	0	0	0	1	88%	DO

Dari **Tabel 2.4** proses pertama yang dilakukan untuk membuat pohon keputusan adalah dengan memilih atribut pertama sebagai *root node*.

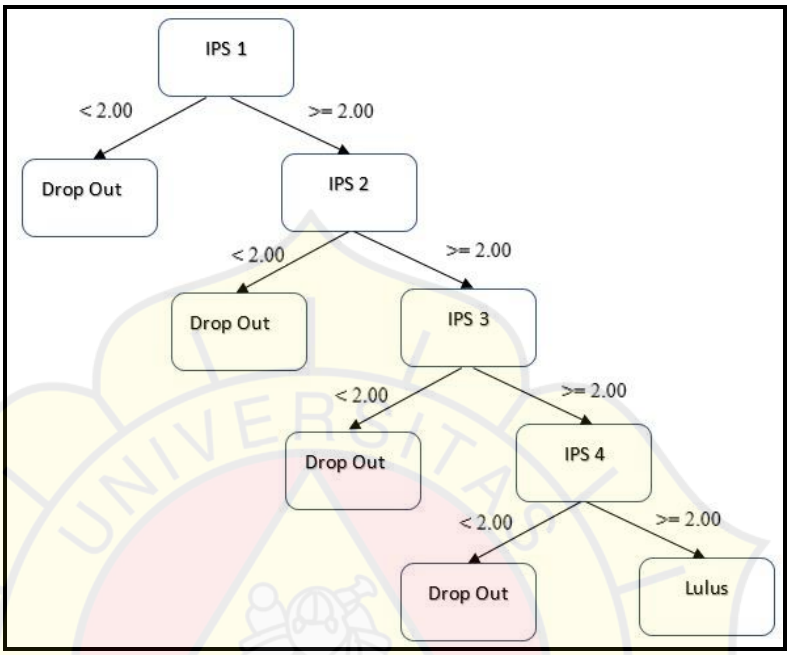
Berdasarkan perhitungan sebelumnya, atribut IPS 1 dipilih karena memiliki *Gain Ratio* tertinggi. *Root node* dari pohon keputusan ini adalah IPS1. Proses selanjutnya adalah membagi dataset berdasarkan nilai IPS1. Jika IPS1 >= 2.0, data akan dikelompokkan ke satu cabang, dan jika IPS1 < 2.0, data akan dikelompokkan ke cabang lainnya. Kelompok IPS1 >= 2.0 akan dipecah lebih lanjut untuk memprediksi kelulusan mahasiswa.

Dari **Tabel 2.4** tersebut terdapat *entropy* dengan jumlah mahasiswa yang memiliki IPS 1 >= 2 ada 6 mahasiswa, mahasiswa yang memiliki IPS 1 < 2 ada 4 mahasiswa. Dari ke 6 mahasiswa yang memiliki IPS 1 >= 2 terdapat 5 mahasiswa Lulus dan 1 mahasiswa *Drop Out*.

Untuk kelompok dengan IPS1 ≥ 2.0, dihitung menggunakan *entropy* dengan formula: $-\left(\frac{5}{6}\log_2\frac{5}{6}+\frac{1}{6}\log_2\frac{1}{6}\right) \approx 0.65$. Dengan menggunakan perhitungan yang sama pada *Split Information* menunjukan bahwa dari 6 mahasiswa, 5 lulus da 1 tidak lulus, maka *entropy* untuk kelulusannya yaitu: $-\left(\frac{5}{6}\log_2\frac{5}{6}+\frac{1}{6}\log_2\frac{1}{6}\right) \approx 0.65$. Untuk atribut IPS2, nilai *Split Information* adalah 1, dengan nilai gain sebesar -0.35 (0.65 – 1) dan *Gain Ratio* sekitar 0.65 atau $\left(\frac{0.65}{1}\right)$.

Pada atribut tagihan semester 1, *entropy*-nya adalah 0.65 dengan perhitungan: $-\left(\frac{1}{6}\log_2\frac{1}{6}+\frac{5}{6}\log_2\frac{5}{6}\right) \approx 0.65$. Nilai *Split Information* juga 1, dengan nilai *gain* sebesar -0.35 (0.65-1) dan *Gain Ratio* sekitar 0.65$\left(\frac{0.65}{1}\right)$. Sementara itu, untuk kelompok dengan IPS1 < 3.0, semua mahasiswa (4 orang) tidak lulus, sehingga *entropy* kelompok ini adalah 0. Karena semua

mahasiswa dalam kelompok ini adalah "Tidak Lulus", tidak perlu melakukan pembagian lebih lanjut. Berikut merupakan stuktur pohon keputusan berdasarkan studi kasus penelitian ini.



Gambar 2.5 Pohon Keputusan Studi Kasus

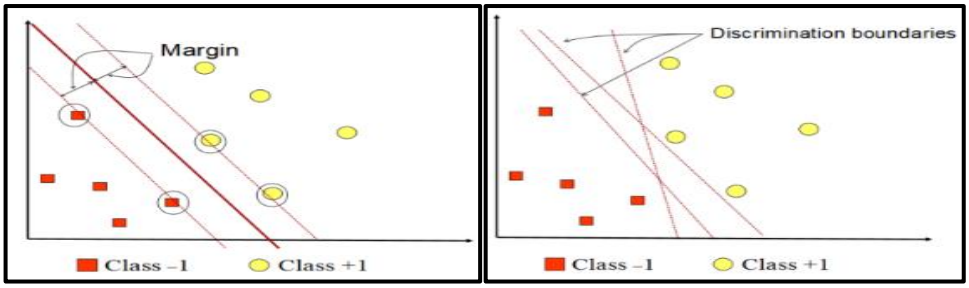
2.5. Metode Support Vector Machine

Support Vector Machine (SVM) merupakan salah satu metode klasifikasi yang digunakan untuk menemukan *hyperplane* optimasi sebagai pemisah data dari berbagai kelas. Metode ini meminimalkan kesalahan klasifikasi serta memaksimalkan margin geometrisnya. Hambatan yang muncul dalam klasifikasi adalah berusaha membedakan sederet data berdasarkan dari kelas yang berbeda. Pada sistem pembelajaran *Support Vector Machine (SVM)*, memanfaatkan ruang hipotesis dalam fungsi linear disuatu fitur yang besar untuk melakukan klasifikasi. Metode ini dilatih dengan menggunakan algoritma pembelajaran berdasarkan teori dan diterapkan dalam *learning bias*

dari teori pembelajaran statistik. Dalam metode ini juga terdapat konsep SVM yang mencari hyperplane atau pemisah di antara beberapa fungsi pemisah yang tak terbatas. Pemisah *(hyperplane)* terbaik di antara kedua kelas yang didapat dengan mencari ukuran margin serta mendapatkan titik maksimumnya.

SVM menggunakan trik kernel yang digunakan untuk memetakan sampel latihan berdasarkan ruang *input* ke ruang fitur yang berdimensi tinggi. Metode *SVM* membentuk metode untuk mengklasifikasikan data, baik secara linier maupun non-linear. Pada konsep metode ini, dapat didefinisikan secara sederhana sebagai cara untuk menemukan *hyperplane* terbaik yang digunakan untuk pemisah pada dua kelas di ruang *input*. **Gambar 2.6** dapat diilustrasikan pola yang terdapat dalam dua kelas, seperti +1 dan -1. Pola yang mencakup pada kelas -1 ditandai warna merah dengan bentuk kotak, sedangkan pola kelas +1 ditandai dengan warna kuning berbentuk lingkaran. Permasalahan klasifikasi dapat diuraikan dengan cara menentukan garis *hyperplane* atau sebagai pemisah di antara dua kelompok. Pada **Gambar 2.5** terdapat alternatif garis yang memisahkan atau *hyperplane* terbaik yang terletak diantara dua kelas dapat ditemukan dengan mengukur *margin hyperplane* serta pola yang mendekati masing-masing kelas. Pola yang mendekati ini disebut *Support Vector*. Garis solid yang berada di **Gambar 2.6** juga memperlihatkan *hyperplane* terbaik yaitu berada tepat di tengah-tengah kedua kelas, sedangkan titik merah dan kuning yang terletak di dalam lingkaran hitam merupakan *Support Vector*. Dalam mencari lokasi *hyperplane*, ini merupakan salah satu langkah pembelajaran dalam metode *SVM*.

Di bawah ini merupakan gambar *hyperplane SVM* terbaik yang ditentukan dalam kelas -1 dan +1



Gambar 2.6 Menentukan Hyperplane SVM terbaik di antara kelas -1 dan +1

Tahap pertama dalam algoritma SVM yaitu menjelaskan persamaan pada hyperplane pemisah dapat menggunakan persamaan berikut:

$$w.X + b = 0$$

Dengan keterangan:

w merupakan bobot vector, $w = \{w_1, w_2, \ldots, w_n\}$;

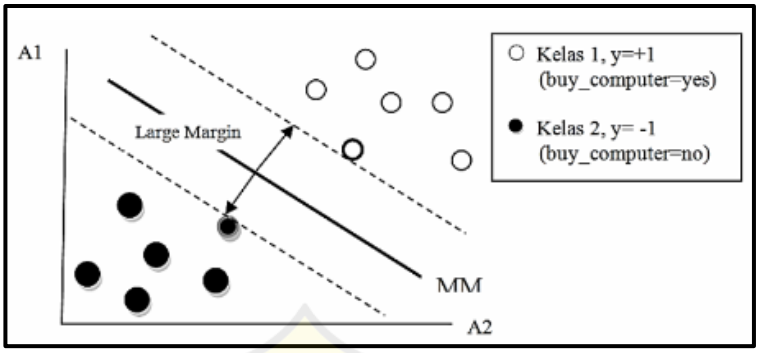
b merupakan skalar yang disebut juga dengan bias, yang berdasarkan pada atribut A_1, A_2 dan jika b ditetapkan sebagai suatu bobot yang menambahkan w_0

Maka persamaan dalam hyperplane atau pemisah dapat didefinisikan dengan persamaan berikut:

$w_o + w_1 x_1 + w_2 x_2 = 0$

Setelah persamaan didefinisikan, nilai x_1 dan x_2 yang tersebut dapat dimasukkan ke dalam persamaan untuk menemukan bobot w_1, w_2, dan w_o atau b. Garis pemisah diantara kedua kelas data menggunakan margin maksimum yang dapat diilustrasikan pada **Gambar 2.7**.

Di bawah ini terdapat **Gambar 2.7** yang merupakan jarak pemisah antara dua kelas data.



Gambar 2.7 Pemisah antara dua kelas data dengan margin maksimum

Gambar di atas mendefinisikan bahwa metode *SVM* mencari *hyperplane* atau garis pemisah maksimum, yaitu *hyperplane* yang terdiri dari jarak maksimum antara *tuple* perlatihan terdekat. Dalam *Support Vector* menunjukkan dengan batasan tebal berdasarkan titik *tuple*. Maka dari itu, masing-masing titik yang terletak di atas *hyperplane* pemisah yang memenuhi persamaan berikut.

$w_o + w_1x_1 + w_2x_2 > 0$

Sedangkan, titik yang berada di bawah hyperlane pemisah yang memenuhi rumus yang terdapat persamaan dibawah ini.

$w_o + w_1x_1 + w_2x_2 < 0$

Berdasarkan dua kondisi di atas, mendapatkan dua persamaan *hyperplane*, seperti yang berada dalam persamaan berikut.

$H_1: w_0 + w_1x_1 + w_2x_2 \geq 0$ untuk $y_1 = +1$

$H_2: w_0 + w_1x_1 + w_2x_2 \leq 0$ untuk $y_1 = -1$

Maka dari itu, masing-masing tuple yang terletak di atas H_1 terdapat kelas +1, dan masing masing *tuple* yang terletak di bawah H_2 yang terdapat kelas

-1. Model *SVM* ini merumuskan dengan menggunakan teknik matematika yaitu formulasi Lagrangian. Berdasarkan formulasi *Lagrangian* ini *Maksimum Margin Hyperplane* (MMH) yang didefenisikan kembali sebagai batas keputusan (*decision boundary*) dengan persamaan di bawah ini.

$$d(X^T) = \sum_{i=1}^{l} y_i \, a_i X_i X^T + b_0$$

y_i merupakan label kelas dari *support vector* $X_i X^T$ adalah suatu tuple test a_i dan b_o merupakan parameter numerik yang ditetapkan dengan cara otomatis dari optimalisasi algoritma *SVM* dan l yang merupakan total *support vector*.

2.6 Pengembangan Aplikasi

Pada bagian ini menjelaskan mengenai pengembangan aplikasi yang mencakup beberapa aspek mulai dari pengertian sistem hingga implementasi teknis berdasarkan penelitian in.

2.6.1 Pengertian sistem

Sistem aplikasi yang ingin dikembangkan pada penelitian ini ialah sebuah aplikasi berbasis website. Sistem aplikasi berbasis website ini bertujuan untuk memudahkan pengguna dalam memprediksi mahasiswa yang berpotensi *Drop Out. Website* memiliki definisi yaitu sebuah halaman yang berisikan sejumlah informasi tertentu yang bisa diakses melalui oleh siapa saja, kapan saja dan dimana saja yang terhubung dengan internet. Pemrograman web ialah suatu cara untuk membuat web dengan menuliskan instruksi atau perintah kepada komputer, agar komputer dapat menjalankan fungsi tertentu. Dan saat dijalankan program tersebut, pengguna dapat mengakses melalui web *browser* seperti *Opera, Mozilla, Chrome,* dan lain sebagainya.

2.6.2 Bahasa Pemrograman

Bahasa pemrograman yang dipakai dalam penelitian ini adalah *Python*. *Python* merupakan bahasa pemrograman yang populer di kalangan pemula hingga para ahli peneliti di dunia. *Python* diciptakan oleh *Guido van Rossum* dan diumumkan pada tahun 1991. Kini, melalui laman https://www.python.org/ dan tersedia di beberapa platform *python* dapat diunduh secara gratis (Khadir, 2019). *Python* merupakan salah satu bahasa pemrograman yang didukung oleh platform *Jupyter Notebook*. *Jupyter* ialah sebuah platform yang digunakan sebagai eksekusi kode langsung melalui peranti lunak atau web. *Jupyter* juga salah satu alat untuk analisis data yang banyak dipakai oleh para ilmuwan data. (Raharjo, 2021)

2.6.3 Unified Modelling Language (UML)

Unified Modelling Language (UML) ialah sebuah Bahasa berbasis grafik serta gambar yang dimanfaatkan untuk memvisualisasikan serta mencatat sistem perangkat lunak yang dikembangkan berbasis *Object-Oriented*. *UML* bukan hanya bahasa pemrograman visual, namun berinteraksi dengan Bahasa pemrograman mencakup *JAVA*, c++, *Visual Basic*, serta database *object-oriented*.

Desain merupakan merupakan salah satu proses dari *SDLC* atau *Software Development Life Cycle*. Tujuan dari desain tersebut adalah untuk memastikan perangkat lunak yang dikembangkan dapat memenuhi kebutuhan dari pengguna. Maka dari itu, langkah penting untuk pengembangan perangkat lunak adalah desain. Dalam tahap mendesain perangkat lunak, perlu melakukan perubahan berdasarkan kebutuhan pengguna dengan fungsional ataupun non-fungsional ke

dalam model yang akurat. (Sumirat, L. P., Cahyono, D., Kristyawan, Y., & Kacung, 2023)

2.6.4 Use Case Diagram

Dalam diagram ini menunjukkan hubungan antara sistem dan aktor. Jenis hubungan yang dilakukan antara pengguna sistem dengan sistem yang dapat digambarkan melalui *use case diagram*.

Tabel 2.5 Use case diagram

Simbol	Keterangan
	Aktor : mempresentasikan peran orang, sistem yang lain atau alat saat berinteraksi dengan use case
	use case: gambaran interaksi antara sistem dengan aktor
	Association: abstraksi dari penghubung antara aktor dengan use case
	generalisasi: memperlihatkan spesialisasi aktor agar bisa berpartisipasi dengan use case
	include: memperlihatkan bahwa suatu use case secara keseluruhan merupakan fungsionalitas dari use case lainnya

	extend: memperlihatkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi

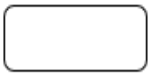
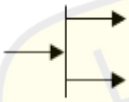
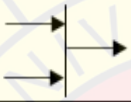
2.6.5 Activity Diagram

Activity Diagram dirancang untuk mendefinisikan alur dari beragam aktivitas dalam suatu sistem yang sedang dibangun, mulai dari tahap awal lalu melakukan proses keputusan yang akan terjadi kemudian melakukan tahap akhir dari alur tersebut. Dalam diagram ini juga menyatakan tahap paralel yang bisa saja terjadi pada langkah-langkah eksekusi. *Activity Diagram* merupakan bentuk *state diagram* tertentu yang merupakan suatu *action* dan transisi yang berasal dari alur sebelumnya.

Activity Diagram menunjukan suatu langkah-langkah dari suatu sistem. Dalam suatu *use case* terdapat paling sedikit satu *activity diagram* yang ada. Diagram ini digunakan untuk urutan alur dalam proses bisnis. Berikut **Tabel 2.6** yang terdapat berbagai simbol di dalam *Activity Diagram*.

Di bawah ini merupakan **Tabel 2.6** yang berisikan simbol-simbol *activity diagram*.

Tabel 2.6 Simbol Activity Diagram

Simbol	Keterangan
	Start Point
	End Point
	Activities
	Fork (Percabangan)
	Join (Penggabungan)
	Decision (Melakukan pilihan)
Swinlane	Suatu cara untuk mengelompokkan simbol-simbol activity berdasarkan actor nya didalam urutan yang sama.

2.7 Paper Terkait

Di bawah ini merupakan *paper* terkait yang mencakup beberapa penelitian yang dilakukan sebelumnya.

Tabel 2.7 Penelitian Sebelumnya

No	Judul	Pengarang & Tahun	Perbandingan
1.	Wira Yuda, Darmawan Tuti, Lim Sheih Yee, Susanti	Penerapan Data Mining untuk klasifikasi Kelulusan Mahasiswa Tepat Waktu Menggunakan Metode Random Forest	Penelitian ini menggunakan variabel klasifikasi berdasarkan kategori IPK dan jumlah sks yang ditempuh oleh mahasiswa di PDPT STMIK Amik Riau. Dengan menggunakan algoritma random forest dan variable yang digunakan, diperoleh tingkat akurasi sebesar 98%
2.	Emy Haryatmi, Sheila Pramita Hervianti	Penerapan Algoritma Support Vector Machine Untuk Model Prediksi Kelulusan Mahasiswa Tepat Waktu	Penelitian ini menerapkan algoritma SVM dan CRISP-DM (Cross Industry Standard Process for Data Mining) untuk menghasilkan model prediksi kelulusan mahasiswa tepat waktu pada Fakultas Teknik Universitas Swasta di Indonesia. Hasil pengujian kelompok pertama dengan jumlah data training sebanyak 90% dan data testing sebanyak 10% menunjukkan bahwa algoritma SVM memberikan nilai akurasi yang sangat baik yaitu 94,4%.
3.	Nur Mahar Aji, Vihi Atina, Nugroho Arif Sudibyo	PEMODELAN PREDIKSI KELULUSAN MAHASISWA DENGAN METODE NAÏVE BAYES DI UNIBA	Penelitian ini memprediksi kelulusan mahasiswa tepat waktu ataupun terlambat berupa data jenis kelamin, status mahasiswa, status menikah, umur, Indeks Prestasi Semester (IPS) dari semester 1-8, dan Indeks Prestasi Kumulatif (IPK)

			dengan menggunakan metode naïve bayes untuk melakukan akurasi dari Naïve Bayes cukup tinggi, yakni mencapai 85%
4.	Fitria Rahmadayanti, Inda Anggraini	Penerapan Metode Decision Tree Dalam Menentukan Kelulusan Mahasiswa	Penelitian ini adalah menghasilkan sistem prediksi penentuan kelulusan mahasiswa tepat waktu dengan metode Decision Tree dan metode Rapid Application Develoment (RAD) pada Sekolah Tinggi Teknologi Pagaralam dengan nilai akurasi 73,19 %