BAB II KAJIAN LITERATUR

KAJIAN LITERATUR

2.1. Tinjauan Pustaka

Bagian ini merupakan landasan teori yang berisi teori-teori pendukung penelitian beserta pekerjaan yang sudah pernah dilakukan oleh peneliti sebelumnya untuk penyelesaian penelitian yang akan dilakukan.

2.1.1. Data Mining

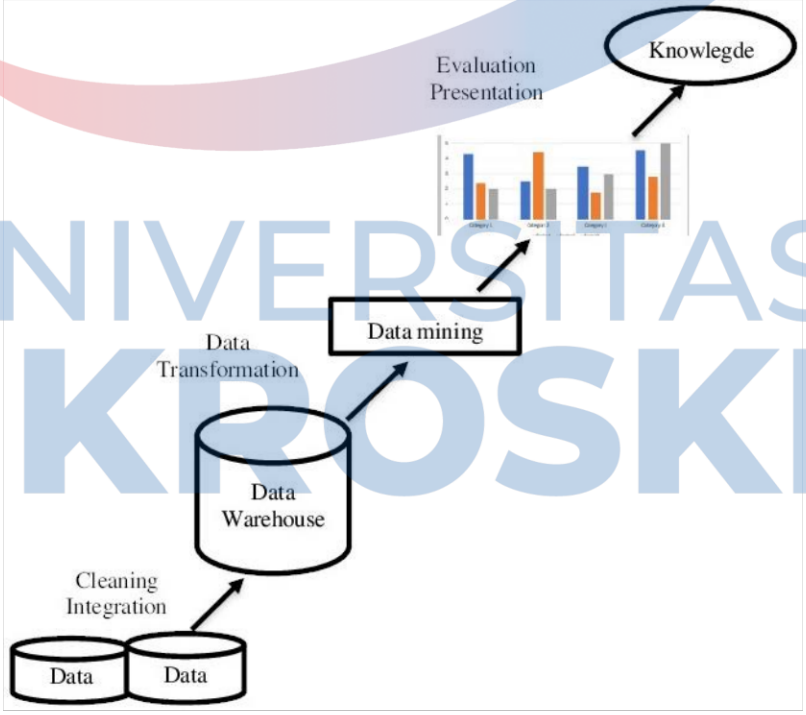
Data mining merupakan proses penggalian dan pencarian pengetahuan dan informasi yang bermanfaat dengan menggunakan algoritma/metode/teknik tertentu sesuai dengan pengetahuan atau informasi yang dicari (Buulolo, 2020). (Vulandari, 2017) mengatakan data mining adalah proses untuk menggali nilai tambah berupa informasi yang selama ini tidak dapat diketahui secara manual dari suatu kumpulan data.

Menurut (Buulolo, 2020), data mining adalah salah satu ilmu yang berkembang sangat pesat, dan perkembangan tersebut disebabkan oleh faktor-faktor, seperti semakin tingginya kesadaran akan pentingnya data, semakin tingginya pemanfaatan output dari hasil pengolahan data dalam berbagai bidang contohnya bidang bisnis, perkembangan kumpulan data yang begitu cepat, peningkatan akses internet baik melalui navigasi *web* ataupun melalui *smartphone*, perkembangan *hardware* dan *software* khususnya yang berhubungan dengan data mining, perkembangan yang begitu cepat dalam bidang komputasi komputer, dan media penyimpanan yang semakin besar dengan harga yang semakin terjangkau.

Data mining memberikan manfaat bagi kehidupan manusia, seperti manfaat komersial dan manfaat dalam bidang keilmuan (Vulandari, 2017). Data banyak digunakan dalam berbagai bidang salah satunya dalam bisnis, pembuatan kebijakan dan pengambilan keputusan (Buulolo, 2020). Data mining berfungsi sebagai penolong dalam pengambilan keputusan karena setiap data transaksi disimpan ke

dalam basis data dan data mining akan menemukan pola algoritma apa yang sesuai digunakan dalam pencarian data (Chouat and Irawan, 2018).

Data mining merupakan salah satu rangkaian *Knowledge Discovery in Database* (KDD). (Buulolo, 2020) mengatakan, tahap awal proses KDD adalah persiapan dan pemilihan data. Kemudian data akan melalui tahap preprocessing untuk dilakukan pembersihan. Setelah itu, data akan ditransformasi agar sesuai dengan algoritma/teknik/metode yang digunakan. Lalu masuk ke tahap data mining yaitu proses mengali dan mencari pengetahuan dan informasi menggunakan algoritma/teknik/metode yang telah ditetapkan. Kemudian pengetahuan atau informasi yang dihasilkan akan ditampilkan ke dalam bentuk yang mudah dimengerti oleh pihak yang berkepentingan. Lalu pengetahuan dan informasi tersebut diimplementasikan sesuai dengan manfaat/kegunaan pengetahuan atau informasi tersebut.



Gambar II-1 Tahapan Proses KDD
Sumber: (Buulolo, 2020)

Data mining memiliki fungsi-fungsi yang umum diterapkan yaitu sebagai berikut (Vulandari, 2017).

1. *Association*, adalah proses untuk menemukan aturan asosiasi antara suatu kombinasi item dalam suatu waktu.
2. *Sequence*, adalah sama seperti *association* namun diterapkan lebih dari satu periode.
3. *Clustering*, adalah proses pengelompokkan sejumlah data/objek ke dalam kelompok data sehingga setiap kelompok berisi data yang mirip.
4. *Classification*, adalah untuk dapat memperkirakan kelas dari suatu objek.
5. *Regression*, adalah untuk memetakan data dalam suatu nilai prediksi.
6. *Forecasting*, adalah untuk mengestimasi nilai prediksi berdasarkan berbagai pola di dalam sekumpulan data.
7. *Solution*, adalah untuk menemukan akar masalah dan menyelesaikan masalah dari persoalan bisnis yang dihadapi untuk digunakan sebagai informasi dalam pengambilan keputusan.

Data mining dibagi menjadi dua kategori utama yaitu (Vulandari, 2017):

1. Prediktif, bertujuan untuk melakukan prediksi terhadap nilai dari atribut tertentu berdasarkan pada nilai atribut-atribut lain.
2. Deskriptif, bertujuan untuk menurunkan pola-pola yang merangkum hubungan yang utama dalam data.

2.1.2. Seleksi Fitur

2.1.2.1. Definisi Seleksi Fitur

Seleksi fitur adalah langkah penting dalam tahap *preprocessing* dan konten penelitian dalam *data mining* dan *machine learning* seperti klasifikasi (Wang *et al.*, 2019). Seleksi fitur dapat mengurangi dimensi data dan waktu pelatihan (AlShboul *et al.*, 2018) serta dapat menghilangkan kemungkinan *noise* dan *overfitting* model (Kamalov and Thabtah, 2017).

Seleksi fitur dilakukan untuk mendapatkan fitur-fitur yang relevan, sehingga memotong aturan (*rule*) klasifikasi menjadi lebih pendek (Kisworini, 2020). Kumpulan fitur yang tepat untuk klasifikasi dapat membantu meningkatkan

performa teknik *machine learning* (Thakkar and Lohiya, 2021). Lebih lanjut, (Thakkar and Lohiya, 2021) menjelaskan bahwa mengikutsertakan seleksi fitur dapat memberikan dua keuntungan yaitu dapat membantu mengurangi *curse of dimensionality* dan mengkomputasikan fitur yang penting dapat membantu interpretasi data.

Selain digunakan untuk meningkatkan kinerja klasifikasi, metode seleksi fitur juga dapat membantu memilih fitur yang paling berpengaruh yang dapat menghasilkan pengetahuan unik untuk diskriminasi antar kelas sehingga dapat mengarah pada pengurangan efek merugikan dari ketidakseimbangan kelas pada kinerja klasifikasi (Leevy *et al.*, 2018).

2.1.2.2. Metode Seleksi Fitur

Metode seleksi fitur adalah *wrapper* dan *filter*. *Wrapper based feature selection* memiliki dua bagian yang disebut sebagai pencarian dan evaluasi. Komponen pencarian menghasilkan pengaturan parameter yang kemudian dievaluasi menggunakan komponen evaluasi. Algoritma ini berinteraksi dengan pengklasifikasi untuk mengevaluasi kegunaan fitur sehingga menghasilkan subset fitur yang lebih baik. Metode *wrapper* ini terdiri dari dua yaitu *forward selection* dan *backward elimination*. *Forward selection* dimulai dengan satu set fitur *null* kemudian mengevaluasi fitur satu per satu pada setiap iterasi. Sedangkan metode *backward elimination* dimulai dengan melibatkan seluruh fitur yang tersedia kemudian menghapus fitur satu per satu di setiap iterasi. Meskipun banyak yang menggunakan, namun *wrapper based feature selection* lebih lambat dibandingkan dengan metode *filter based feature selection* (Balasaraswathi, Sugumaran and Hamid, 2017)

2.1.2.3. Chi-Square Feature Selection

Chi-square feature selection adalah salah satu metode *filter based feature selection* yang merupakan bagian dari *supervised feature selection*. Seleksi fitur

dilakukan untuk mengidentifikasi *subset* atribut yang optimal (Thaseen, Kumar and Ahmad, 2019). *Chi-square* menggunakan uji independensi untuk menilai apakah fitur tersebut tidak bergantung pada label kelas (Li *et al.*, 2017). *Chi-Square* mengevaluasi independensi kemandirian dua peristiwa (kemunculan fitur dan kemunculan kelas) untuk sekumpulan data tertentu (Thakkar and Lohiya, 2021).

$$X^2 = \frac{(O-E)^2}{E} \quad \text{.....(2.1)}$$

Untuk menyeleksi fitur, skor X^2 optimal yang dipilih. Jika suatu fitur memiliki skor X^2 yang rendah, maka fitur tersebut tidak bergantung pada kelas target yang menyiratkan bahwa fitur tersebut tidak informatif untuk mengklasifikasikan sampel data (Thakkar and Lohiya, 2021).

Seleksi fitur dengan metode *chi-square* telah diterapkan dalam berbagai bidang penelitian seperti dalam memprediksi loyalitas konsumen (Sulistiani and Tjahyanto, 2017), prediksi kebangkrutan (Rustam, Nadhifa and Acar, 2018), klasifikasi *credit scoring* (Trivedi, 2020), dan *intrusion detection* (Thaseen, Kumar and Ahmad, 2019; Thakkar and Lohiya, 2021)

2.1.3. Ketidakseimbangan Kelas

2.1.3.1. Definisi Ketidakseimbangan Kelas

Salah satu masalah dalam klasifikasi data mining adalah ketika kelas tidak diwakili secara seimbang. Masalah ini disebut ketidakseimbangan dataset (Japkowicz, 2013). Dataset terbagi menjadi kelas mayoritas dan kelas minoritas, dimana kelas mayoritas memiliki jumlah *instance* yang lebih banyak dibanding kelas minoritas. Setiap dataset dengan distribusi yang tidak merata antara kelas mayoritas dan minoritas dapat dianggap memiliki ketidakseimbangan kelas, dan dalam aplikasi dunia nyata, tingkat keparahan ketidakseimbangan kelas dapat bervariasi dari minor hingga parah (tinggi atau ekstrim) (Leevy *et al.*, 2018).

Masalah ketidakseimbangan kelas menjadi tantangan dalam proses klasifikasi dan menarik perhatian banyak peneliti (Haixiang *et al.*, 2017).

Permasalahan ketidakseimbangan kelas (*class imbalance*) membuat proses belajar *classifier* sulit (Japkowicz, 2013). Selain itu, ketidakseimbangan kelas akan mempengaruhi kualitas data dalam hal kinerja klasifikasi (Gao, Khoshgoftaar and Wald, 2014) dan menyebabkan kinerja klasifikasi menjadi tidak optimal (Nurmasani and Pristyanto, 2021). Dalam proses klasifikasi, kelas minoritas sering salah diklasifikasikan, karena *machine learning* memprioritaskan kelas mayoritas dan mengabaikan kelas minoritas (Santoso, Wibowo and Himawati, 2019).

Ketika kelas tidak seimbang, *learners* biasanya akan mengklasifikasikan kelompok mayoritas secara berlebihan karena meningkatnya probabilitas. Akibatnya, kejadian yang termasuk dalam kelompok minoritas lebih sering salah diklasifikasikan daripada yang termasuk dalam kelompok mayoritas (Johnson and Khoshgoftaar, 2019). Jika tingkat ketidakseimbangan kelas untuk kelas mayoritas ekstrim, maka pengklasifikasi dapat menghasilkan akurasi prediksi keseluruhan yang tinggi karena model tersebut kemungkinan besar memprediksi sebagian besar sampel sebagai milik kelas mayoritas (Leevy *et al.*, 2018).

2.1.3.2. Pendekatan dalam Menangani Ketidakseimbangan Kelas

Salah satu cara untuk menangani dataset yang tidak seimbang adalah pendekatan level data yaitu *resampling*, seperti metode *random undersampling* dan metode *random oversampling*. Metode *random undersampling* menyeimbangkan dataset dengan cara menghilangkan anggota dari kelas mayoritas secara acak. Metode ini terkadang dapat memberikan hasil yang tidak diinginkan karena sifatnya yang acak (Yildirim, 2016). Sementara pada metode *random oversampling*, penyeimbangan data dilakukan dengan cara menaikkan *instance* minoritas secara acak. Menyeimbangkan dataset dengan meningkatkan jumlah *instance* kelas dapat meningkatkan kinerja dari kelas minoritas, dan dengan demikian dapat meningkatkan *precision* dan *recall* dari kelas-kelas ini (Goodfellow, Bengio and Courville, 2016). Metode ini dapat meningkatkan *precision* dan *recall*, namun terlihat terjadi penurunan pada akurasi pengklasifikasi (Sakar *et al.*, 2019).

Pada pendekatan menggabungkan atau memasangkan (*ensemble*) metode, ada dua algoritma *ensemble-learning* paling populer, yaitu *boosting* dan *bagging* (Yap *et al.*, 2014). Bagging meminimalkan varians prediktif dengan menghasilkan beberapa set pelatihan dari set data yang diberikan, dengan pengklasifikasi yang dihasilkan untuk setiap set pelatihan dan kemudian model individualnya digabungkan untuk klasifikasi akhir (Leevy *et al.*, 2018). *Boosting* juga menggunakan beberapa set pelatihan dari kumpulan data yang diberikan, dan setelah secara berulang menetapkan bobot yang berbeda untuk setiap klasifikasi berdasarkan kesalahan klasifikasi, pendekatan berbobot yang menggabungkan hasil klasifikasi individu menghasilkan klasifikasi akhir (Leevy *et al.*, 2018). Algoritma *boosting* telah dilaporkan sebagai meta-teknik untuk mengatasi masalah ketidakseimbangan kelas (*class imbalance*) (Sun *et al.*, 2007). *Boosting* telah menunjukkan dapat meningkatkan kinerja pengklasifikasi dalam banyak situasi, termasuk ketika data tidak seimbang (Seiffert *et al.*, 2008).

2.1.3.3. Adaptive Boosting (AdaBoost)

Algoritma *adaptive boosting* (*adaboost*) membangun pengklasifikasi kuat dengan cara mengombinasikannya dengan sejumlah pengklasifikasi sederhana (lemah) (Mulyati, Yulianti and Saifudin, 2017). Cara kerja algoritma adaboost adalah memberikan bobot yang sama pada setiap sampel pada awalnya. Setelah setiap klasifikasi, bobot hasil yang benar berkurang, dan bobot hasil yang salah bertambah. Proses ini diulangi hingga mencapai ambang batas atau jumlah siklus maksimum (Wu *et al.*, 2008; Yap *et al.*, 2014; Hao *et al.*, 2020).

1. Suatu dataset pelatihan dengan N sampel memiliki dua kelas dengan label $y \in \{0,1\}$
2. Tetapkan bobot awal yang sama untuk setiap sampel dalam set pelatihan.
   $w_i^1 = \frac{1}{N}, i = 1,2, \dots, N$ .....(2.2)
3. Untuk setiap iterasi $t = 1,2, \dots, T$

a. Pilih secara acak kumpulan data dengan N sampel dari kumpulan pelatihan asli menggunakan *weighted resampling*. Peluang sampel untuk dipilih terkait dengan bobotnya. Sampel dengan bobot yang lebih tinggi memiliki kemungkinan yang lebih tinggi untuk dipilih.
b. Dapatkan *learner*, *f(x)* (model prediktif atau pengklasifikasi) dari kumpulan data yang disampel ulang.
c. Terapkan *learner f(x)* ke set data pelatihan asli. Jika sampel salah diklasifikasikan, jika salah = 1, jika tidak = 0.
d. Hitung jumlah *weighted errors* dari semua sampel pelatihan.

$error^t = \sum_{i=1}^{N}(w_i^t \times error_i^t)$ .....(2.3)

e. Hitung indeks kepercayaan learner f(x):

$\alpha^t = \frac{1}{2}\ln\left(\frac{1-error^t}{error^t}\right)$ .....(2.4)

Indeks kepercayaan dari *learner f(x)* tergantung pada *weighted error*.

f. Perbarui bobot semua sampel pelatihan asli:

$$w_i^{t+1} = w_i^t \times \begin{cases} \exp(-\alpha^t) & \text{untuk menurunkan bobot} \\ \exp(\alpha^t) & \text{untuk menaikkan bobot} \end{cases}$$

.....(2.5)

Jika sampel diklasifikasikan dengan benar, bobotnya menurun, sedangkan bobot untuk sampel yang salah diklasifikasikan meningkat.

g. Kemudian, lakukan normalisasi bobot,

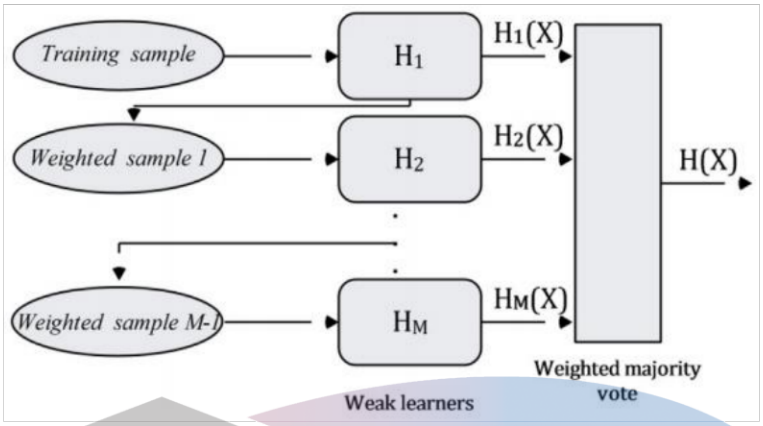
$w_i^t = \frac{w_i^t}{\sum_i^N w_i^t}$ .....(2.6)

sehingga,

$\sum_i^N w_i^{t+1} = 1$ .....(2.7)

h. $T = t + 1$, jika $error < 0.5$, atau $t < T$, ulangi langkah a-g; jika tidak, berhenti dan $T = t - 1$.
i. Setelah T iterasi, $t = 1,2,\ldots,T$, terdapat model prediksi $f^t(x)$, $t = 1,2,\ldots T$. Prediksi akhir untuk kasus j, diperoleh dari prediksi gabungan model *T* menggunakan pendekatan *voting*:

$y_j = sign\sum_{t=1}^{T}\alpha^t f^t(x)$ .....(2.8)



Gambar II-2 Algoritma AdaBoost
Sumber: (Shahraki, Abbasi and Haugen, 2020)

Penerapan *adaptive boosting* (*adaboost*) diterapkan pada beberapa penelitian di bawah ini.

1. (Mulyati, Yulianti and Saifudin, 2017) mengombinasikan *adaboost* dengan *naïve bayes* untuk klasifikasi *churn* pelanggan industri telekomunikasi.
2. (Sudarto, Zarlis and Sirait, 2016) mengombinasikan adaboost dengan C4.5 untuk klasifikasi kelulusan mahasiswa.
3. (Sarkar, Verma and Maiti, 2018) mengombinasikan adaboost dengan C5.0 untuk klasifikasi *occupational incidents*.
4. (Mazini, Shirazi and Mahdavi, 2019) menerapkan *adaboost* untuk mengevaluasi dan mengklasifikasikan fitur untuk *intrusion detection*.
5. (Chen *et al.*, 2021) mengombinasikan *adaboost* dengan KNN untuk mendemonstrasikan pemahaman emosi dinamis dari robot dalam interaksi manusia-robot.

2.1.4. Klasifikasi

2.1.4.1. Definisi Klasifikasi

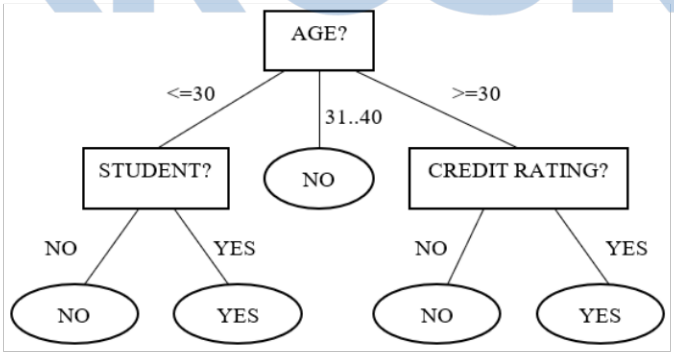
Dalam data mining ada begitu banyak algoritma/metode/teknik penggalian atau pencarian pengetahuan atau informasi. Setiap algoritma/metode/teknik tersebut mempunyai fungsi dan tujuan yang berbeda-beda. Salah satu fungsi dan

tujuan itu adalah klasifikasi. (Vulandari, 2017) menjelaskan proses klasifikasi data memiliki dua tahapan, yang pertama adalah *learning*. Yang dimaksud dengan *learning* adalah *training data* dianalisa dengan menggunakan sebuah algoritma klasifikasi. Dan yang kedua adalah *classification* yaitu pada tahap ini *test data* digunakan untuk mengestimasi ketepatan dari *classification rules*.

Algoritma klasifikasi memberikan hasil berbeda untuk dataset atau masalah yang berbeda. Tidak ada algoritma klasifikasi khusus yang dapat menyelesaikan masalah yang ada. Algoritma klasifikasi yang dianggap sebagai solusi terbaik untuk memecahkan masalah belum tentu dapat berfungsi di masalah yang lain (Çiğşar and Ünal, 2019).

2.1.4.2. Pohon Keputusan

Metode klasifikasi yang paling terkenal adalah pohon keputusan. Interpretasi pengklasifikasi pohon keputusan adalah sederhana dan dapat dengan mudah dipahami dengan penjelasan singkat (Breiman *et al.*, 2017). Dengan pohon keputusan, manusia dapat dengan mudah mengidentifikasi dan melihat hubungan antara faktor-faktor yang mempengaruhi suatu masalah dan dapat mencari penyelesaian terbaik dalam memperhitungkan faktor tersebut (Vulandari, 2017). Pohon keputusan dapat menangani *overfitting*, menangani atribut kontinu, memilih atribut yang tepat untuk pemilihan atribut, menangani data pelatihan dengan nilai atribut yang hilang, dan meningkatkan efisiensi komputasi (Quinlan, 1993).



Gambar II-3 Contoh Pohon Keputusan
Sumber: (Vulandari, 2017)

2.1.4.3. Algoritma C5.0

C5.0 adalah salah satu algoritma data mining yang khususnya diterapkan pada algoritma pohon keputusan. Algoritma C5.0 termasuk dalam kategori pohon klasifikasi (Yu *et al.*, 2018). Ini adalah versi lanjutan dari pengklasifikasi C4.5 dengan kinerja *superior*. Algoritma C5.0 lebih baik daripada C4.5 pada akurasi, kecepatan, dan memori (Ross Quinlan, 2017). C5.0 memberikan tingkat akurasi terbaik dan waktu eksekusi yang lebih sedikit dibandingkan dengan algoritma klasifikasi lain (Rajeswari and Suthendran, 2019).

Berikut ini adalah algoritma C5.0 (Joloudari *et al.*, 2020)

a. *Gain ratio* digunakan sebagai dasar pembentukan *node* atau akar dan cabang pohon keputusan. *Gain ratio* dapat dihitung sebagai berikut.

$$GainRatio(K,C) = \frac{Gain(K,C)}{SplitInfo(K,C)} \quad \text{.....(2.9)}$$

b. Berdasarkan persamaan (2.9), $SpliInfo(K,C)$ dan $Gain(K,C)$ dihitung sebagai berikut.

$$SplitInfo(K,C) = Info_{Entropy}\left(\frac{|C_1|}{|C|}, \dots, \frac{|C_i|}{|C|}\right) \quad \text{.....(2.10)}$$

$$Gain(K,C) = Info_{Entropy}(K) - Info_{Gain}(K,C) \quad \text{.....(2.11)}$$

Dimana K adalah jumlah fitur dan C_i adalah partisi dari C yang diturunkan dari nilai K.

c. Dengan demikian, dirumuskan $Info_{Gain}(K,C)$ dan $Info_{Entropy}(K)$ sebagai berikut.

$$Info_{Entropy}(K) = -\sum_{i=1}^{N \in C_i} P_i \log_2 P_i \quad \text{.....(2.12)}$$

$$Info_{Gain}(K,C) = -\sum_{i=1}^{N \in C_i} P_i \times Info_{Entropy}(K_i) \quad \text{.....(2.13)}$$

d. Menurut persamaan (2.12) dan (2.13), P dihitung sebagai berikut.

$$P = \left(\frac{|C_1|}{|S|}, \frac{|C_2|}{|S|}, \dots, \frac{|C_i|}{|S|}\right) \quad \text{.....(2.14)}$$

Dimana $|S|$ adalah jumlah contoh dalam himpunan S dan P adalah distribusi probabilitas partisi $(C1, C2, \dots, Ci)$

Algoritma klasifikasi dengan metode C5.0 telah diterapkan dalam beberapa bidang penelitian seperti untuk mengidentifikasi faktor dan diagnosis kekambuhan kanker ovarium (Tseng *et al.*, 2017), klasifikasi *occupational incidents* (Sarkar, Verma and Maiti, 2018), indeks standar polusi udara (Putra and Sitanggang, 2020), prediksi efisensi operasional bank (Appiahene, Missah and Najim, 2020), dan *credit scoring* (Trivedi, 2020).

2.1.5. Evaluasi

2.1.5.1. Confusion Matrix

Kinerja dari model klasifikasi dievaluasi dengan *confusion matrix*. Tabel II-1 merupakan *confusion matrix* yang menggambarkan *true positive* (TP), *false negative* (FN), *false positive* (FP), dan *true negative* (TN). (Dipto *et al.*, 2020) menjelaskan secara singkat apa yang dimaksud dengan TP, FN, FP, dan TN. TP adalah *instance* positif yang benar diklasifikasikan sebagai positif. FN adalah *instance* positif yang diklasifikasikan sebagai negatif. FP adalah *instance* negatif yang diklasifikasikan sebagai positif. Sedangkan TN adalah *instance* negatif yang benar diklasifikasikan sebagai negatif.

Tabel II-1 Confusion Matrix

		Actual	
		Positive	**Negative**
Predicted	**Positive**	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Berdasarkan nilai yang tercantum pada *confusion matrix*, beberapa evaluasi dapat didefinisikan sebagai berikut.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad \text{.....(2.15)}$$

Akurasi (*accuracy*) adalah metrik yang paling umum digunakan ketika mengevaluasi hasil klasifikasi. Namun, ketika bekerja dengan dataset yang tidak

seimbang, akurasi saja tidak cukup karena nilai yang dihasilkan didominasi oleh kelompok mayoritas (*majority class*), yaitu kelas negatif (Johnson and Khoshgoftaar, 2019).

$$Precision = \frac{TP}{TP+FP} \quad \text{.....(2.16)}$$

Presisi (*precision*) adalah metrik yang menggambarkan tingkat keakuratan antara data yang diminta dengan hasil yang diberikan oleh model. Metrik ini sensitif terhadap ketidakseimbangan kelas karena mempertimbangkan jumlah sampel negatif yang diklasifikasikan sebagai positif (FP). Presisi saja tidak cukup untuk mengevaluasi hasil, karena tidak memberikan wawasan tentang jumlah sampel dari kelompok positif yang diklasifikasikan sebagai negatif (FN) (Johnson and Khoshgoftaar, 2019).

$$Recall/Sensitivity\ (TPR) = \frac{TP}{TP+FN} \quad \text{.....(2.17)}$$

Recall mengukur persentase kasus positif yang benar-benar positif. Recall tidak terpengaruh oleh ketidakseimbangan karena hanya bergantung pada kelas yang positif (Johnson and Khoshgoftaar, 2019).

$$Selectivity/Specificity\ (TNR) = \frac{TN}{TN+FP} \quad \text{.....(2.18)}$$

Selectivity mengukur persentase kasus negatif yang benar-benar negatif (Johnson and Khoshgoftaar, 2019). *Selectivity* juga tidak terpengaruh oleh ketidakseimbangan karena hanya bergantung pada kelas yang negatif.

$$F1\ score = 2\frac{precision*recall}{precision+recall} \quad \text{.....(2.19)}$$

F1 score adalah rata-rata harmonik *precision* dan *recall* (Chicco and Jurman, 2020).

2.1.6. Penelitian Terdahulu

Bagian ini merupakan bagian yang berisi beberapa penelitian terdahulu yang telah dilakukan dengan dataset yang tidak seimbang menggunakan berbagai algoritma klasifikasi.

(Tseng *et al.*, 2017) menggunakan algoritma C5.0, MARS, RF, ELM, dan SVM untuk mengidentifikasi faktor dan diagnosis kekambuhan kanker ovarium. Hasil penelitian menunjukkan bahwa model C5.0 lebih unggul dalam memprediksi kambuhnya kanker ovarium dan C5.0 memberikan kinerja yang lebih baik dalam hal akurasi. (Putra and Sitanggang, 2020) menggunakan metode C5.0 dan *random forest* untuk klasifikasi indeks standar polusi udara, memberikan hasil akurasi C5.0 mencapai 100% pada dataset tahun 2017 dan rata-rata akurasi C5.0 secara keseluruhan lebih tinggi dibanding *random forest*. (Appiahene, Missah and Najim, 2020) menggunakan algoritma C5.0, *random forest*, dan *neural network* untuk prediksi efisiensi operasional bank dengan validasi *10-fold cross-valdiation*, menyatakan bahwa C5.0 memberikan model prediksi terbaik dengan akurasi mencapai 100% diikuti dengan *random forest* 98.5% dan *neural network* 86.6%. (Wang and Wu, 2020) melakukan penelitian menggunakan algoritma C5.0, random trees, KNN, logistic, dan bayesian serta metode *undersampling* pada dataset *macao mobile phone*, hasil penelitian menunjukkan bahwa algoritma C5.0 memiliki tingkat akurasi tertinggi.

(Sun and Liu, 2017) menggunakan algoritma KNN, SVM, DT (C4.5) dan NB, serta metode *SMOTE* untuk menyeimbangkan data, hasil penelitian menunjukkan bahwa metode tersebut dapat meningkatkan kinerja klasifikasi. (Yildirim, 2016) menggunakan algoritma *radial basis function* (RBF *networks*), IBK (*K-nearest neighbour*), ID3, dan *random tree* untuk prediksi hasil efek sampling *albendazole* serta metode *resample*, *SMOTE*, *spread sub sample*, *stratified removed fold* untuk mengatasi ketidakseimbangan kelas. Hasil penelitian menunjukkan bahwa ID3 dengan *resample* memiliki hasil akurasi yang lebih tinggi daripada yang lain. (Sakar *et al.*, 2019) menggunakan algoritma *random forest* dan *naïve bayes*, serta menerapkan metode *oversampling* untuk menyeimbangkan

dataset *Online Shoppers Purchasing Intention*. Hasil penelitian menunjukkan bahwa metode *oversampling* menaikkan TPR dan *f1 score* namun menurunkan akurasi seluruh algoritma klasifikasi. (Fernández *et al.*, 2017) menggunakan algoritma *decision tree* dan *random forest* dengan metode ROS *(random oversampling)*, SMOTE dan RUS *(random undersampling)*, menunjukkan bahwa ROS dan RUS menghasilkan hasil klasifikasi yang lebih baik dibanding SMOTE. (Mulyati, Yulianti and Saifudin, 2017) melakukan klasifikasi *churn* pelanggan industri telekomunikasi dengan algoritma *naïve bayes* dan metode ROS, RUS, *AdaBoost*. Hasil penelitian menunjukkan bahwa kombinasi *naïve bayes* dengan ROS dan *AdaBoost* menghasilkan AUC tertinggi. (Sudarto, Zarlis and Sirait, 2016) menggunakan algoritma C4.*5* dan metode DBFS (*density based feature selection*) dan *AdaBoost* (*adaptive boosting*) untuk klasifikasi kelulusan mahasiswa dengan validasi 5, 10, 20, dan *30-fold cross-validation*, hasil penelitian menunjukkan bahwa DBFS dan *AdaBoost* dapat meningkatkan kinerja klasifikasi. (Hao *et al.*, 2020) menggunakan algoritma *boosting* (*C5.0*, *GBM*), *bagging* (*bagged CART*, *random forest*), dan *stacking* (*stack.glm*, *stack.rf*) untuk *Bank Direct Marketing Analysis* dengan validasi *10-fold cross-validation*, menunjukkan bahwa model *C5.0* berbasis algoritma *boosting* memiliki tingkat akurasi, nilai Kappa dan nilai AUC tertinggi di antara enam model. (Sarkar, Verma and Maiti, 2018) menggunakan algoritma CART dan C5.0 untuk klasifikasi *occupational incidents* dengan validasi *10-fold cross-validation* serta *adaptive boosting* (*adaboost*) untuk meningkatkan akurasi. Hasil penelitian menunjukkan kombinasi C5.0 dan *adaboost* menghasilkan akurasi yang lebih tinggi dibanding yang lain.

(Thakkar and Lohiya, 2021) melakukan penelitian *intrusion detection* menggunakan metode DT, RF, KNN, LR, NB, SVM, dan ANN, serta metode seleksi fitur *chi-square*, IG, dan RFE dengan validasi *10-fold cross-validation*, hasil penelitian menunjukkan bahwa seleksi fitur dapat meningkatkan kinerja model. (Thaseen, Kumar and Ahmad, 2019) yang menerapkan SVM, MNB, LPBoost untuk *intrusion detection*, dan menyeleksi dataset dengan metode *chi-square*, menyatakan bahwa *chi-square* dapat mempertimbangkan fitur yang bergantung pada label kelas dan akurasi klasifikasi dapat mencapai 100%. (Sulistiani and

Tjahyanto, 2017) menggunakan algoritma *random forest* untuk prediksi loyalitas konsumen dengan validasi *10-fold cross-validation*. Dataset diseleksi dengan metode *chi-square*, *gain ratio*, dan *informasion gain*, manunjukkan bahwa *chi-square* dapat meningkatkan akurasi klasifikasi dan menjadi metode terbaik dalam seleksi fitur dibanding metode yang lain. (Trivedi, 2020) menggunakan algoritma *bayesian*, SVM, C5.0 dan RF untuk klasifikasi *credit scoring* dengan validasi *10-fold cross-validation*, dataset diseleksi dengan metode *chi-square*, *information gain*, dan *gain ratio*, menunjukkan bahwa kombinasi C5.0 dan *chi-square* adalah yang terbaik.

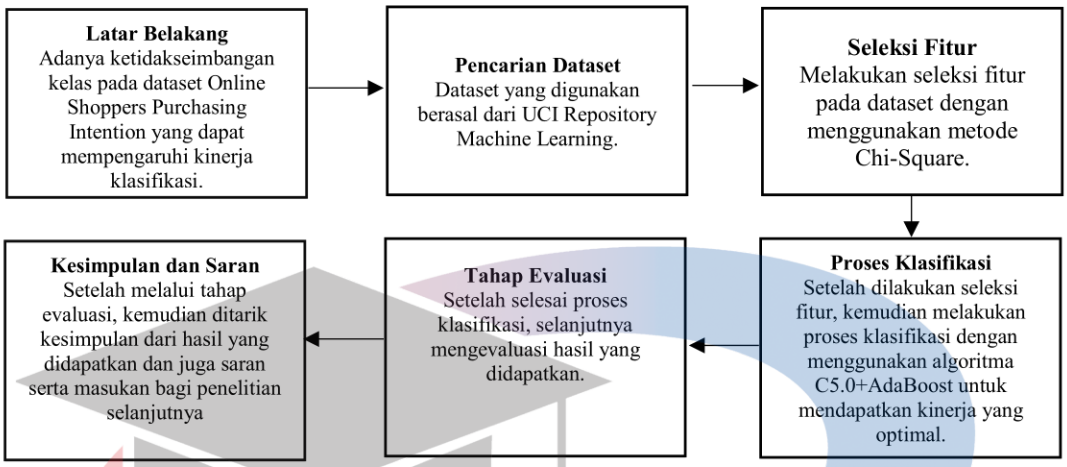
2.2. Kerangka Konsep/Pola Pikir Pemecahan Masalah

Salah satu masalah dalam klasifikasi adalah ketidakseimbangan kelas, dimana kelas mayoritas memiliki jumlah *instance* yang lebih banyak dibanding kelas minoritas. (*Online Shoppers Purchasing Intention Dataset*, 2018) adalah salah satu contoh dataset yang tidak seimbang, dimana hanya sebagian kecil yang berakhir dengan pembelian, dan sebagian besarnya tidak.

Masalah ketidakseimbangan kelas membuat proses belajar *classifier* menjadi sulit dan akan mempengaruhi kinerja klasifikasi. Dalam proses klasifikasi dengan dataset yang tidak seimbang, kelas minoritas sering salah diklasifikasikan, karena algoritma klasifikasi memprioritaskan kelas mayoritas dan cenderung mengabaikan kelas minoritas. Hal ini menjadi tantangan dalam proses klasifikasi karena mengklasifikasikan kelas minoritas sama pentingnya dengan kelas mayoritas.

Dalam penelitian ini, pendekatan yang dilakukan untuk mengatasi masalah ketidakseimbangan kelas adalah melalui seleksi fitur dan metode *adaptive boosting* (*adaboost*). Metode seleksi fitur yang digunakan adalah metode *filter* yaitu *chi-square feature selection*. Setelah melalui tahap seleksi, fitur-fitur yang terpilih akan dimasukkan ke dalam *classfier* menggunakan metode algoritma C5.0 dengan dan tanpa *adaptive boosting* (*adaboost*). Kinerja dari model yang dihasilkan akan

dievaluasi menggunakan *confusion matrix* untuk mengetahui *accuracy*, *precision*, *recall/sensitivity/TPR*, *selectivity/specificity/TNR*, dan *f1 score*.



Gambar II-4 Kerangka Konsep Pemecahan Masalah