BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian terdahulu menjadi fondasi penting untuk pengembangan studi baru, menyediakan konteks dan pemahaman mendalam tentang suatu topik. Penelitian terdahulu mencakup informasi dasar untuk mengidentifikasi *gap* penelitian yang masih ada. Tabel 2.1 merupakan ringkasan penelitian terdahulu yang telah dikumpulkan.

Tabel 2. 1 Penelitian Terdahulu

Judul Artikel Jurnal	Hasil	Kesimpulan	Hasil Klasifikasi	Elemen yang diadopsi
Application of Data Mining for Rainfall Prediction Classification in Australia with Decision Tree Algorithm and C5.0 Algorithm [6]	Penelitian dilakukan dengan membandingkan efektivitas metode validasi *10-cross vold* dan metode *split* 80:20 pada kasus prediksi hujan di Australia menggunakan 2 algoritma yaitu *Decision Tree* dan C5.0 *Rule-Based Model*. Hasil terbaik didapatkan oleh algoritma *Decision Tree* dengan metode *10-Cross Fold Validation* 87.35% dan	Penelitian ini membuktikan bahwa penggunaan *10-Cross Fold Validation* lebih baik dari metode *split* 80:20 pada algoritma *Decision Tree* dan C5.0 *Rule-Based Model* untuk memprediksi hujan di Australia. Algoritma yang terbaik di antara keduanya adalah *Decision Tree* dengan akurasi	*Decision Tree Splitting Data* akurasi 85.37%, *Decision Tree 10 Cross Validation* akurasi 85.37%.	*Decision Tree* dan metode *data splitting*

	penggunaan metode *split* 80:20 juga menunjukkan hasil 85.37%	pada metode *10-Cross Fold Validation* 87.35% dan penggunaan metode *split* 80:20 juga menunjukkan hasil 85.37%		
The Right Sentiment Analysis Method of Indonesian Tourism in Social Media Twitter [11]	Penelitian melakukan analisis sentimen publik terhadap kota Bali berdasarkan data yang tersedia pada media sosial *Twitter*. Model dilatih menggunakan *Naïve Bayes, Neural Network, Support Vector Machine,* dan *Decision Tree.* Hasil algoritma *Decision Tree* memiliki akurasi 72.94%, presisi 71.48% dan presisi 96.37%	Penelitian ini melakukan klasifikasi analisis sentimen publik terhadap kota Bali menggunakan beberapa algoritma *machine learning*. Hasilnya menggunakan algoritma *Decision Tree* memiliki akurasi 72.94%, presisi 71.48% dan sensitivitas 96.37%. Nilai sensitivitas dari algoritma *Decision Tree* merupakan yang tertinggi di antara semua algoritma klasifikasi lain yang digunakan	*Decision Tree* akurasi 72.94%, presisi 71.48% dan sensitivitas 96.37%.	*Decision Tree*

Particle Swarm Optimization Feature Selection for Breast Cancer Prediction [12]	Penelitian melakukan klasifikasi menggunakan algoritma klasifikasi *Naïve Bayes, Decision Tree, K-Nearest Neighbors* dan *Logistic Regression* untuk memprediksi kanker payudara dengan dan tanpa seleksi fitur menggunakan *GA* dan *PSO*. Hasil *Decision Tree* tanpa menggunakan *PSO* akurasi 93.50%, presisi 94.97%, dan *recall* 94.62%. Setelah menggunakan seleksi fitur menggunakan *PSO* akurasi meningkat menjadi 94.56%, presisi naik menjadi 96.25%, dan sensitivitas menjadi 95.01%.	Penggunaan *PSO* sebagai seleksi fitur untuk mengklasifikasikan kanker payudara mampu meningkatkan semua aspek kinerja metrik. Peningkatan akurasi menunjukkan bahwa menggunakan seleksi fitur dengan *PSO* mampu meningkatkan kemampuan model dan peningkatan presisi serta sensitivitas menunjukkan bahwa model lebih baik dalam mengidentifikasi kelas yang benar dan salah.	*Decision Tree* tanpa seleksi fitur akurasi 93.50%, presisi 94.97%, dan sensitivitas 94.62%, *Decision Tree* dengan seleksi fitur *PSO* akurasi 94.56%, presisi 96.25%, sensitivitas 95.01%.	*Decision Tree* dan seleksi fitur menggunakan *PSO*
Enhanced Model for Prediction and Classification of Cardiovascul	Penelitian membandingkan dua model prediksi penyakit kardiovaskular menggunakan model algoritma	Penelitian ini membuktikan bahwa seleksi fitur dengan algoritma *Particle swarm optimization*	*Decision Tree* akurasi 85.24% dengan waktu pemrosesan 0.11 detik, *Decision Tree*	*Decision Tree* dan seleksi fitur menggunakan *PSO*

ar Disease using Decision Tree with Particle Swarm Optimization [7]	*Decision Tree* dan model *Decision Tree* yang dilatih dengan fitur hasil dari seleksi fitur menggunakan algoritma *Particle Swarm optimization.* Algoritma *Decision Tree* Menunjukkan akurasi sebesar 85.24% dengan waktu pemrosesan 0.11, menggunakan semua 14 atribut dari *dataset* dan *Decision Tree* dengan seleksi fitur algoritma *PSO* menghasilkan akurasi sebesar 83.61% dengan waktu pemrosesan 0.02 detik, dengan hanya menggunakan 7 atribut terpilih berdasarkan *PSO*	pada algoritma *Decision Tree* mempengaruhi efektivitas dan efisiensi prediksi. Walaupun model dengan *PSO* memiliki akurasi yang sedikit lebih rendah (83.61%) dibandingkan dengan model tanpa *PSO* (85.24%) namun model *PSO* menunjukkan keunggulan yang signifikan dalam waktu pembangunan model yang hanya memerlukan 0.02 detik	dengan seleksi fitur menggunakan *PSO* akurasi 83.61%, waktu pemrosesan 0.02 detik.	
Data Splitting Techniques to Reduce False-Positive and False-Negative Cases in	Penelitian menggunakan beberapa *data splitting* rasio untuk memprediksi kanker payudara. Dataset yang	Penelitian ini menunjukkan penggunaan *Data Splitting* rasio 70% untuk training dan 30% untuk testing	Hasilnya algoritma *DF, LR,* dan *NN* memiliki akurasi diatas 98% dan algoritma SVM	*Data Splitting* rasio 70:30

Breast Cancer Prediction [16]	digunakan adalah *Wisconsin Breast* Cancer dengan 4 algoritma klasifikasi yaitu SVM, Logistic Regression, Decision Forest, dan Neural Network. Hasilnya algoritma DF, LR, dan NN memiliki akurasi diatas 98% dan algoritma SVM mendapatkan akurasi diatas 97%	menunjukkan hasil yang akurat dan seimbang	mendapatkan akurasi diatas 97%	
An investigation of XGBoost-based algorithm for breast cancer classification [14]	Penelitian mengembangkan sistem pakar kalsifiaksi kanker payudara menggunakan teknik pembelajaran mesin yang inovatif. Teknik yang digunakan melibatkan penggunaan jaringan pra-latih *DenseNet201* untuk belajar dari gambar histologi yang kemudia dilakukan klasifikasi menggunakan *Xgboost* berdasrkan	Model yang ditawarkan mampu menghindari *overfitting* dan bias dalam model. Metode memiliki keunggulan dalam memproses dan mengklasifikasikan berbagai jenis tumor payudara.	*Xgboost* akurasi rata-rata 97%.	*Xgboost*

	vektor fitur yang diekstrak mencapai akurasi rata-rata 97%			
Breast Cancer Classification using XGBoost [15]	Penelitian menggunakan *Xgboost* untuk mengklasifikasikan sel kanker payudara sebagai benigna atau maligna. Pelatihan menunjukkan bahwa model mampi mengenali pola dalam nuklei sel payudara dan menentukkan fitur yang penting dengan hasil akurasi 94.74%	Penggunaan *Xgboost* dalam penelitian menunjukkan efektivitas yang signifikan dengan menekankan fitur-fitur yang penting dalam membedakan karakteristik sel kanker. Pendekatan menggunakan *Xgboost* membantu dan menguatkan model prediksi.	*Xgboost* akurasi 94.74%.	*Xgboost*
An optimized Xgboost technique for accurate brain tumor detection using feature selection and image segmentation [8]	Penelitian ini menggunakan teknik canggih untuk mengolah gambar medis untuk tumor otak yang lebih akurat. Pertama menggunakan metode *Clahe* untuk meningkatkan kualitas gambar, kedua segmentasi gambar, ketiga menggunakan seleksi fitur	Penelitian menunjukkan bahwa integrasi dari penggunaan metode *CLAHE*, segmentasi *K-means*, dan seleksi fitur menggunakan algoritma *PSO* pada 3 algoritma klasifikasi dapat meningkatkan akurasi, presisi,	*Xgboost* dengan seleksi fitur menggunakan *PSO* akurasi 97%, presisi 98%, dan sensitivitas 97 %.	*Xgboost* dan seleksi fitur menggunakan *PSO*

	dengan algoritma *PSO* pada tiga algoritma klasifikasi digunakan yaitu *Xgboost*, Naïve Bayes, dan ID3. Hasilnya yang paling bagus adalah *Xgboost* dengan nilai akurasi 97%, presisi 98%, dan sensitivitas 97 %	sensitivitas dalam diagnosis tumor otak khususnya pada penggunaan *PSO-Xgboost* yang mampu berperan pentin untuk meningkatkan keakuratan diagnosis dengan mengurangi kemungkinan kesalahan manusia		
Specific Emitter Identification Based on ACO-Xgboost Feature selection Jianjun [9]	Penelitian ini menguji dan membandingkan model dengan seleksi fitur *ACO* pada *Xgboost* dengan algoritma klasifikasi lain seperti *Decision Tree*, *Random forest*, dan *Gradient Boosted Decision Tree*s untuk identifikasi emitor spesifik hasilnya pada *dataset* original *ACO-Xgboost* mampu meningkatkan akurasi pada tiga *dataset* berbeda	Penelitian ini menunjukkan bahwa model dengan seleksi fitur menggunakan *ACO* pada *Xgboost* memberikan hasil akurasi dan pemilihan fitur yang lebih efisien dalam identifikasi emitor spesifik. Model tidak hanya menunjukkan nilai akurasi yang tinggi namun juga efisiensi dengan jumlah	*Xgboost* dengan seleksi fitur menggunakan *ACO dataset* original akurasi 98.90%, *dataset* 10db 73.00%, dan *dataset* 5db 59.30%.	*Xgboost* dan seleksi fitur menggunakan *ACO*

	yaitu *dataset original, dataset 10db, dataset 5db* antara 0.20%-3.53% dibandingkan dengan algoritma lain dan metode ini berhasil mendapatkan *subset* fitur dengan jimlah yang relatif kecil.	fitur yang relatif lebih kecil sehingga terjadi keseimbangan.		
Fracture Type Identification Using Extra tree classifier [10]	Penelitian mengembangkan sistem klasifikasi untuk mengidentifikasi jenis, lokasi, dan jenis fraktur tulang menggunakan gambar medis. Algoritma *Extra tree classifier* yang dioptimalkan oleh berbagai teknik pengolahan gambar menunjukkan kinerja yang baik dengan akurasi 92.13%, presisi 91.85% dan recall 90.96%	Penelitian melakukan pengolahan gambar untuk klasifikasi tipe patah tulang. Algoritma *Extra Tree* dan berbagai teknik pengolahan gambar seperti penggunaan *filter* multipel, deteksi tepi *canny* dan metode ekstrasi fitur seperti *GWT*, *HOG*, dan *GLCM* berhasil melakukan klasifikasi tipe patah tulang dengan akurasi diatas 90%	*Extra Tree Classifier* akurasi 92.13%, presisi 91.85% dan sensitivitas 90.96%.	*Extra Tree Classifier*

Penelitian [6] dilakukan dengan membandingkan efektivitas dari metode *10-Cross Fold Validation* dengan metode *data splitting* 80 % : 20 % pada kasus

prediksi hujan di Australia menggunakan 2 algoritma yaitu *Decision Tree* dan C5.0 *Rule-Based Model*. Hasilnya menunjukkan bahwa metode *10-Cross Fold Validation* memberikan performa yang lebih unggul dibandingkan dengan metode *split* 80 : 20. Akurasi menggunakan 10 - *Cross Fold Validation* pada algoritma *Decision Tree* mendapatkan hasil paling tinggi dengan angka 87.35 % dibandingkan dengan metode *split* yang hasilnya 85.37% dalam memprediksi hujan di Australia. Penelitian ini menunjukkan bahwa penggunaan algoritma Decision Tree untuk melakukan.

Penelitian [11] dilakukan dengan menganalisis sentimen publik terhadap kota Bali berdasarkan data dari media sosial *Twitter* menggunakan beberapa *machine learning* yaitu *Naïve Bayes, Neural Network, Support Vector Machine,* dan *Decision Tree*. Dalam mengklasifikasikan sentimen publik terhadap kota bali algoritma *Decision Tree* menunjukkan kinerja yang cukup dengan akurasi sebesar 72.94%, presisi 71.48%, dan recall tinggi yaitu 96.37%. Nilai sensitivitas menggunakan algoritma *Decision Tree* merupakan yang tertinggi bila dibandingkan dengan algoritma klasifikasi yang lain.

Penelitian [12] dilakukan dengan menggunakan algoritma klasifikasi *Naïve Bayes, Decision tree, K-Nearest Neighbors,* dan *Logistic Regression* untuk memprediksi kanker payudara, dengan dan tanpa seleksi fitur menggunakan *Genetic Algorithm* dan *PSO*. Hasil untuk *Decision Tree* tanpa seleksi fitur menggunakan *PSO* menunjukkan akurasi 93.50%, presisi 94.97%, dan sensitivitas 94.62%. Hasil *Decision Tree* menerapkan seleksi fitur menggunakan *PSO* meningkatkan akurasi menjadi 94.56%, presisi naik menjadi 96.25%, dan sensitivitas menjadi 95.01%. Penggunaan *PSO* sebagai metode seleksi fitur meningkatkan semua aspek kinerja metrik, menunjukkan bahwa seleksi fitur menggunakan *PSO* dapat meningkatkan kemampuan model dalam mengidentifikasi kelas yang benar dan lebih akurat.

Penelitian [16] Penelitian menggunakan beberapa rasio *data splitting* untuk memprediksi kanker payudara. *Dataset* yang digunakan adalah *Wisconsin Breast Cancer* dengan 4 algoritma klasifikasi yaitu *SVM, Logistic Regression, Decision Forest,* dan *Neural Network*. Hasilnya algoritma *DF, LR,* dan *NN* memiliki akurasi di atas 98% dan algoritma *SVM* mendapatkan akurasi di atas 97%. Penelitian ini menunjukkan penggunaan *Data Splitting* rasio 70% untuk *training* dan 30% untuk *testing* menunjukkan hasil yang akurat, seimbang dan bisa di andalkan.

Penelitian [14] dilakukan dengan mengembangkan sistem pakar klasifikasi kanker payudara dengan menggunakan teknik pembelajaran mesin yang inovatif, melibatkan pemanfaatan jaringan pralatih *DenseNet201* untuk belajar fitur dari gambar histologi dan dilakukan klasifikasi menggunakan algoritma *Xgboost* berdasarkan fitur yang berhasil diekstrak. Hasil akurasi rata-rata menggunakan *Xgboost* adalah 97%. Model yang ditawarkan memiliki keunggulan dalam menghindari *overfitting* dan bias serta efektif dalam memproses dan mengklasifikasikan berbagai jenis tumor payudara.

Penelitian [15] dilakukan dengan menggunakan *Xgboost* untuk mengklasifikasikan sel kanker payudara sebagai benigna atau maligna. Pelatihan menggunakan algoritma *Xgboost* mampu mengenali pola dalam nuklei sel payudara dan menentukan fitur penting dengan hasil akurasi 97.74%. Penggunaan *Xgboost* menunjukkan efektivitas yang signifikan dalam menekankan fitur-fitur penting dalam karakteristik sel kanker dan menunjukkan potensi dalam meningkatkan akurasi dalam prediksi kanker payudara.

Penelitian [7] dilakukan dengan menguji dua model prediksi penyakit kardiovaskular dengan menggunakan algoritma *Decision Tree*. Model pertama menggunakan 14 fitur yang tersedia pada *dataset* tanpa seleksi fitur dan model kedua menggunakan 7 fitur yang telah melalui seleksi fitur menggunakan algoritma *Particle swarm optimization* (*PSO*). Hasilnya model pertama yang tanpa melalui

seleksi fitur dan hanya menggunakan *Decision Tree* saja mendapat akurasi 85.24% dan waktu pemrosesan 0.11 detik, sementara model kedua yang melalui proses seleksi fitur menggunakan *PSO* pada algoritma *Decision Tree* mendapatkan akurasi 83.61% dengan waktu pemrosesan 0.02 detik. Hasil dari kedua model menunjukkan bahwa walaupun akurasi model dengan seleksi fitur menggunakan *PSO* menunjukkan hasil akurasi yang lebih rendah namun seleksi fitur menawarkan efisiensi waktu pembangunan model yang signifikan.

Penelitian [8] dilakukan dengan mengaplikasikan teknik canggih pengolahan gambar medis untuk meningkatkan akurasi diagnosis dari tumor otak. Metode pertama yang digunakan adalah *CLAHE* yang digunakan untuk meningkatkan kualitas gambar, metode kedua adalah segmentasi gambar dengan menggunakan metode *K-means*, dan metode ketiga adalah menggunakan seleksi fitur dengan algoritma *PSO* yang diterapkan pada tiga algoritma klasifikasi yaitu *Xgboost*, *Naïve Bayes*, dan *ID3*. Hasilnya dari ketiga algoritma yang menunjukkan hasil terbaik adalah *PSO-Xgboost* dengan hasil akurasi 97%, presisi 98%, dan sensitivitas 97%.

Penelitian [9] dilakukan dengan menguji dan membandingkan efektivitas model *ACO-Xgboost* menggunakan seleksi fitur *ACO* dengan algoritma klasifikasi lain seperti *Decision Tree*, *Random forest*, dan *Gradient Boosted Decision Tree*s. Hasilnya penelitian ini membuktikan bahwa model *ACO-Xgboost* dapat meningkatkan akurasi sebesar 0.20% hingga 3.53% dibandingkan dengan algoritma lain dan memilih jumlah fitur yang paling sedikit sehingga tidak hanya meningkatkan akurasi namun *ACO-Xgboost* menawarkan sebuah keseimbangan antara akurasi dan efisiensi pemilihan fitur yang optimal.

Penelitian [10] dilakukan dengan mengembangkan sistem klasifikasi yang menggunakan gambar medis untuk mengidentifikasi, lokasi, dan jenis fraktur tulang pada gambar. Algoritma yang dimanfaatkan untuk klasifikasi gambar medis

adalah *Extra tree classifier* yang dioptimalkan dengan berbagai teknik pengolahan gambar seperti penggunaan *filter* multipel, deteksi tepi *canny* dan metode ekstraksi fitur seperti *GWT*, *HOG*, dan *GLCM*. Hasil dari akurasi *Extra tree classifier* menunjukkan kinerja yang sangat baik dengan akurasi 92.13%, presisi 91.85%, dan sensitivitas 90.96%. Penelitian ini menunjukkan bahwa sistem klasifikasi menggunakan *Extra tree classifier* mencapai keseimbangan yang baik dalam mengidentifikasi berbagai jenis fraktur dan tulang berdasarkan gambar medis.

Penelitian sebelumnya telah mengeksplorasi penggunaan algoritma *Decision Tree* untuk klasifikasi dengan teknik pemisahan data, dan seleksi fitur menggunakan *PSO* yang menunjukkan efisiensi waktu. Selain itu, penggunaan metode seleksi fitur *PSO* dan *ACO* pada *Xgboost* terbukti dapat meningkatkan akurasi dan efisiensi pemilihan fitur. Algoritma *Extra tree classifier* yang digunakan untuk mengklasifikasikan gambar medis juga telah menunjukkan kemampuan menghasilkan keseimbangan yang baik antara akurasi, presisi, dan recall, dengan nilai di atas 90%. Pembagian *data splitting* 70 untuk data *training* dan 30 untuk *data testing* juga menunjukkan keseimbangan dan performa yang baik pada algoritma berbasis pohon keputusan. Penelitian ini bertujuan untuk mengklasifikasikan penyakit *stroke* menggunakan ketiga algoritma tersebut *Decision Tree*, *Xgboost*, dan *Extra tree classifier* dan membandingkannya dalam kondisi dengan dan tanpa proses seleksi fitur menggunakan algoritma *PSO* dan *ACO* dengan pembagian *data splitting* 70% untuk *data training* dan 30% untuk *data testing*. Penelitian ini tidak hanya memberikan wawasan mengenai efektivitas masing-masing algoritma dan metode seleksi fitur, tetapi juga memberikan kontribusi signifikan terhadap praktik klinis dalam mendeteksi dan mencegah *stroke* lebih awal.

2.2 Tinjauan Teori

2.2.1 Penyakit Stroke

Penyakit *stroke* didefinisikan sebagai gangguan neurologis akut pada pembuluh darah di otak penyebabnya karena suplai darah ke area otak berhenti dan sel-sel otak mengalami kekurangan oksigen dan mati secara bertahap, dan kecacatan yang akan dialami penderita tergantung kepada seberapa parah area otak yang terkena dampaknya. Penyakit *stroke* dibagi menjadi *stroke* iskemik dan *stroke* hemoragi. *Stroke* hemoragi merupakan *stroke* dengan kondisi pecahnya pembuluh darah yang mengakibatkan terjadi pendarahan di otak, *stroke* jenis ini merupakan *stroke* yang jarang terjadi. *Stroke* iskemik merupakan jenis *stroke* yang paling umum dengan kondisi penghentian aliran darah ke area otak karena terjadi penyempitan atau penyumbatan arteri [3].

2.2.2 Seleksi Fitur

Seleksi fitur adalah teknik *processing* yang digunakan untuk mengidentifikasi fitur kunci dari masalah yang diberikan. Algoritma pembelajaran biasanya menggunakan data yang terdiri dari sampel dan fitur untuk mendefinisikan data. Algoritma pembelajaran memerlukan jumlah sampel yang besar karena bila sampel sedikit maka dapat menyebabkan kapasitas generalisasi. Jumlah sampel yang besar biasanya tidak terdiri dari fitur yang benar-benar dibutuhkan dan akan menimbulkan sulitnya optimasi yang dilakukan oleh enumerasi lengkap dalam ruang produk. Untuk mengatasi masalah ini seleksi fitur merupakan salah satu teknik yang dapat digunakan untuk pengurangan dimensi. Pengurangan dimensi artinya proses menemukan matriks dengan kolom yang sedikit dengan tujuan memilih fitur yang palling relevan, dan menyeleksi fitur yang tidak relevan untuk dibuang [17].

Supervised feature selection biasanya berorientasi pada masalah klasifikasi dengan menggunakan relevansi atau korelasi antara fitur dan label kelas sebagai prinsip dasarnya. Ukuran relevansi digunakan untuk

mengevaluasi pentingnya fitur. Salah satu model yang diusulkan dengan tujuan untuk menemukan pemilihan *subset* menggunakan pendekatan heuristik untuk mengevaluasi efek dari fitur tunggal yang bersesuaian dengan setiap kategori untuk mendapat fitur yang optimal adalah penggunaan *CFS*. Selain *CFS* ada juga model *Relief* yang digunakan pada masalah *multi-class* dengan ide utama mengambil jarak *Euclidean* sebagai indeks korelasi dan selanjutnya akan diberikan bobot pada fitur sesuai dengan seberapa baik mereka dapat membedakan *instance* dari kelas yang berbeda [18].

2.2.3 Swarm Intelligence

Swarm Intelligence adalah cabang kecerdasan buatan yang relatif baru digunakan dengan tujuan untuk memodelkan perilaku kolektif kawanan sosial dialam seperti semut, lebah, dan kawanan burung. *Swarm* merupakan sejumlah besar agen homogen dan yang berinteraksi secara lokal di antara lingkungan mereka sendiri namun tanpa kontrol pusat yang memungkinkan untuk munculnya perilaku menarik global. *Swarm-based algorithm*s yang muncul sebagai keluarga algoritma berbasis populasi terinspirasi oleh observasi terhadap alam yang mampu menghasilkan solusi dengan biaya yang rendah, cepat, dan mampu menyelesaikan masalah kompleks. Interaksi sosial yang terjadi antar individu *swarm* dapat bersifat langsung maupun secara tidak langsung [19].

2.2.4 Shapiro-Wilk

Shapiro-wilk test merupakan uji statistik yang menawarkan metode yang efisien dan terpercaya untuk menguji normalitas, terutama untuk sampel yang lengkap. Uji ini sangat bermanfaat karena banyak prosedur statistik mengasumsikan bahwa data mengikuti distribusi normal dan kesalahan akibat asumsi ini bisa mempengaruhi hasil validitas hasil

analitis yang membuat uji ini penting untuk dilakukan [20]. Rumus persamaan *Shapiro-Wilk Test* ditunjukkan di persamaan (2.1).

$$W = \frac{(\Sigma_{i=1}^{n} a_i y_{(i)})^2}{\Sigma_{i=1}^{n}(y_i - \bar{y})^2} = \frac{b^2}{S^2} \quad (2.1)$$

2.1 Rumus Shapiro Wilk Test

Penjelasan dari rumus persamaan (2.1*) Shapiro-Wilk Test* adalah sebagai berikut:

- W adalah uji statistik *Shapiro-Wilk.*
- n adalah ukuran sampel.
- i adalah indeks atau nomor urut dari elemen
- $y_{(i)}$ adalah observasi ke-i dalam sampel yang telah diurutkan dari kecil ke yang terbesar
- a_i adalah koefisien yang diturunkan nilai ekspektasi order statistik dari distribusi normal baku
- $\bar{y}$ adalah rata-rata sampel
- y_i adalah nilai observasi ke-i dalam sampel

2.2.6 ANOVA

Uji *ANOVA (Analysis of Variance)* adalah metode statistik yang digunakan untuk menguji rata-rata antar kelompok dalam berbagai penelitian eksperimental, termasuk dalam ilmu sosial dan perilaku. Uji *ANOVA* digunakan untuk membuktikan apakah ada perbedaan yang signifikan di antara mereka [21]. Rumus persamaan uji *ANOVA* ditunjukkan di persamaan (2.2) [22]:

$$F = \frac{MS_{between}}{MS_{within}} \quad (2.1)$$

Penjelasan dari rumus persamaan (2.2*)* uji *ANOVA* adalah sebagai berikut:

- F adalah nilai statistik uji *ANOVA.*

- $MS_{between}$ adalah rata-rata variansi antara grup yang mengukur variasi antar rata-rata antar kelompok yang dihitung dari *sum of squares* antara grup dibagi dengan derajat kebebasan antar grup.
- MS_{within} adalah rata-rata variansi dalam grup, yang mengukur variasi dalam kelompok yang dihitung dari *sum of squares* dalam grup dan derajat kebebasan dalam grup.

2.2.5 ADASYN

ADASYN (Adaptive Synthetic Sampling) adalah teknik yang digunakan untuk mengatasi ketidakseimbangan dalam data. Teknik *ADASYN* merupakan penciptaan data sintetis untuk kelas minoritas. *ADASYN* menerapkan distribusi berbobot untuk kelas minoritas berdasarkan tingkat kesulitan pembelajarannya. Artinya data sintetis yang dihasilkan untuk kelas minoritas yang lebih sulit dipelajari lebih banyak dibandingkan dengan contoh yang lebih sulit dipelajari. Tujuannya adalah untuk mengurangi bisa yang disebabkan ketidakseimbangan kelas [23].

2.3 Framework dan Algoritma

2.3.1 CRISP-DM

Cross Industry Standard Process – Data Mining (*CRISP-DM*) merupakan model proses industri-independen untuk *data mining*. *CRISP-DM* terdiri dari enam fase berulang mulai dari *business understanding* sampai *deployment* [24]. Saat ini, siklus *CRISP-DM* merupakan c yang paling banyak digunakan untuk mengerjakan proyek *Data Mining*. Penggunaan metodelogi *CRISP-DM* membantu pengguna untuk membuat kemajuan pertama dibidang *Data Mining* karena metode ini memiliki prosedur yang bertahap dalam penerapannya sehingga proses pengerjaan *Data Mining* menjadi lebih

terstruktur. Enam fase yang ada dalam metodelogi *CRISP-DM* yaitu: [25].

1) *Business Understanding*

   Tahapan *business understanding* Merupakan Tahap pemahaman bisnis dari proyek yang sedang dikerjakan mengandalkan keahlian dari kreativitas analisis. Situasi bisnis perlu dianalisis karena perlu diketahui gambaran tentang sumber daya yang tersedia dan yang dibutuhkan. Pada tahap ini penting untuk mengetahui objek dari bisnis dan bagaimana cara membuat kecocokan tujuan pemodelan untuk tujuan bisnis.

2) *Data Understanding*

   Tahapan *data understanding* merupakan tahap dilakukannya pengumpulan data dari sumber data yang akan digunakan dalam proses *data mining*. Setelah data dikumpulkan data akan dieksplorasi dan dideskripsikan. Pada fase ini penting untuk memeriksa kualitas dari data yang dikumpulkan.

3) *Data Preparation*

   Tahapan *data preparation* merupakan tahap dilakukannya pemilihan data yang sesuai dengan kriteria inklusi dan eksklusi yang telah ditentukan sebelumnya. Saat kualitas data yang didapat buruk maka dapat ditangani dengan membersihkan data. Proses persiapan data dilakukan tergantung pada model yang akan digunakan jadi langkahnya mungkin berbeda dalam setiap kasus.

4) *Modelling*

   Tahapan *modelling* merupakan tahap pemodelan data yang terdiri dari pemilihan teknik pemodelan, membangun kasus uji dan model. Pada tahap pemodelan semua teknik *data mining* dapat digunakan. Secara umum, pilihannya tergantung kepada masalah bisnis dan data dan bagaimana data ingin dijelaskan. Saat

melakukan pembentukan model maka langkah yang harus dilakukan adalah menetapkan parameter untuk menilai model apakah tepat untuk mengevaluasi model terhadap kriteria evaluasi dan memilih yang terbaik.

5) *Evaluation*

Tahapan *Evaluation* merupakan tahap memeriksa hasil terhadap tujuan bisnis yang ditetapkan. Oleh karena itu, hasil harus ditafsirkan dan tindakan lebih lanjut yaitu mendefinisikannya. Proses melakukannya akan ditinjau secara umum.

6) *Deployment*

Tahapan *Deployment* merupakan tahap model akhir dikerahkan. *Deployement* bergantung kepada persyaratan proyek, fase penyebaran dapat menjadi serumit atau sesederhana penerapan dalam model operasi. Fase ini akan menghasilkan rencana penyebaran sehingga nantinya jelas apa yang akan dilakukan pada proses penyebaran.

2.3.2 Decision Tree

Decision Tree merupakan algoritma *machine learning* yang efektif digunakan untuk melakukan klasifikasi dan regresi [26]. Struktur *Decision Tree* serupa dengan pohon yang terdiri dari *node* keputusan dan *node* daun. *Node* keputusan mewakili pernyataan atau kondisi memisahkan data, sementara *node* daun mewakili hasil atau keputusan akhir. Algoritma *Decision Tree* memulai dari *root node* dan memecah data menjadi *sub-set* berdasarkan fitur yang memberikan informasi terbanyak menggunakan metrik seperti *information gain* dan *gini index*. Entropi mengukur ketidakpastian dalam set data dalam konteks pohon keputusan.

Pada pembuatan *Decision Tree, root node* dianggap sebagai titik awal yang akan memuat seluruh *dataset*, kemudian akan dibagi menjadi

bagian yang lebih kecil berdasarkan fitur yang paling ideal untuk meminimalisir impuritas. Proses ini akan terjadi sampai tidak ada lagi data untuk dibagi sehingga terbentuk *node* daun sebagai hasil akhir. Salah satu proses yang bisa dilakukan adalah proses pemangkasan yang digunakan untuk menghilangkan cabang yang tidak diperlukan, mempertahankan akurasi model, dan mengoptimalkan ukuran pohon. Metode yang dilakukan meliputi *cost complecity pruning* dan *error reduction pruning* yang membantu mengurangi risiko *overfitting*. Berikut merupakan *pseudocode* dari algoritma *Decision Tree* pada Tabel 2.2 [27].

Tabel 2. 2 *Pseudocode* of *Decision Tree Algorithm*

GenDecTree(Sample S, Feature F)	
Steps:	
1.	*If stopping_condition(S,F) = true then*
	a. Leaf = createNode()
	b. leafLabel = classify(s)
	c. return leaf
2.	*root = createNode()*
3.	*root_test_condition = findBestSplit(S,F)*
4.	*V={v | v a possible outcomecfroot.test_condition}*
5.	*For each value v ∈ V:*
	a. s_v *= {s | root,test_condition(s) = v and s ∈ S};*
	b. Child = TreeGrowth (s_v*, F);*
	c. Add child as descent of root and label the edge {root →child}as v
6.	*return root*

Sumber: Hambali (2019)

Berikut merupakan penjelasan simbol yang digunakan pada *pseudocode Decision Tree*:

1) S: Himpunan sampel yang digunakan untuk membangun pohon keputusan.
2) F: Himpunan fitur yang tersedia untuk membagi sampel.
3) *stopping_condition(S,F)*: fungsi untuk memeriksa himpunan S memiliki label yang sama atau jumlah sampel yang terlalu sedikit.

4) *Leaf* adalah *node* daun dalam pohon keputusan yang mewakili akhir keputusan.
5) *createNode*: membuat *node* baru dalam pohon keputusan.
6) *classify(s*): mengembalikan label klasifikasi himpunan sampel S.
7) *root*: *node* akar dari pohon keputusan.
8) *findBestSplit(S,F)* fungsi mencari pembagian terbaik dari sampel S berdasarkan fitur F.
9) V: himpunan dari semua nilai yang mungkin untuk kondisi uji di *root.*
10) v: nilai individu dari himpunan V.
11) s_v: *subset* dari s yang memenuhi kondisi.
12) *TreeGrowth* (s_v, F): memanggil kembali fungsi GenDecTree secara rekursif untuk membuat pohon keputusan untuk s_v dengan fitur *F*.
13) *Child*: *node* anak yang dihasilkan dari pemanggilan rekursif.

2.3.3 Xgboost

Xgboost merupakan singkatan dari *Extreme gradient boosting* yang merupakan algoritma yang dirancang untuk melakukan optimasi dengan efisiensi tinggi dalam pembelajaran mesin khususnya untuk model linear dan algoritma pembelajaran pohon [28]. *Xgboost* mendukung berbagai fungsi objektif termasuk *ranking*, klasifikasi, regresi, menawarkan solusi yang fleksibel untuk berbagai jenis data, dan skenario analisis. *Xgboost* mengontrol kompleksitas model untuk mempermudah proses pembelajaran dan mencegah *overfitting*, mendukung paralelisasi dalam pemilihan titik *split* terbaik, sehingga meningkatkan kecepatan untuk waktu pelatihan. *Xgboost* akan menghentikan proses pembangunan pohon ketika sudah didapat hasil prediksi yang cukup baik, hasilnya waktu pemrosesan dapat diturunkan. Algoritma ini menawarkan keunggulan dalam menangani data

yang disusun dalam tabel berdimensi rendah dan interpretasi model yang lebih baik, invariansi data *input* dan kemudahan penyesuaian parameter. Tabel 2.3 merupakan *pseudocode Xgboost* [29].

Tabel 2. 3 Pseudocode Xboost Algorithm

Initialize fo(x)
for k = 1 to M do
compute gk = $\frac{\partial(L(y,f))}{\partial f}$
Compute hk = $\frac{\partial^2(L(y,f))}{\partial f^2}$
Determine the structure by choosing splits with maximized gain
A = $\frac{1}{2}[\frac{G_L^2}{H_L} + \frac{G^2}{H} + \frac{1}{2}]$
Determine the leaf weights w =* - $\frac{G}{H}$
Determine the base learner b(x) = Σ_j^T *wI*
Add trees fk(x) = fk-1(x) + b(x)
end for
Result: f(x) = Σ_k^M *fk(x)*

Sumber: Assawab (2021)

Berikut merupakan penjelasan simbol yang digunakan pada *pseudocode Xgboost*:

1) *fo(x)*: model awal atau prediksi awal.
2) M: jumlah iterasi atau jumlah pohon yang dibuat.
3) K: indeks untuk iterasi yang berjalan dari 1 hingga M.
4) *gk*: gradien dari fungsi *loss* $(L(y, f)$ terhadap prediksi saat ini f dihitung sebagai turunan pertama dari *loss* terhadap prediksi.
5) *hk*: *hessian* dari fungsi *loss* $(L(y, f)$ terhadap prediksi saat ini f dihitung sebagai turunan kedua dari *loss* terhadap prediksi.
6) $(L(y, f)$: fungsi *loss* untuk mengukur kesalahan prediksi f yang dibandingkan dengan nilai sebenarnya y.
7) $\frac{1}{2}[\frac{G_L^2}{H_L} + \frac{G^2}{H} + \frac{1}{2}]$: formula untuk menghitung *gain* dari *split* G_L dan H_L.
8) G_L: jumlah gradien di sisi kiri *split.*
9) H_L: jumlah *hessian* di sisi kiri *split.*

10) G: jumlah total gradien.
11) H: jumlah total *hessian.*
12) *w*:* berat daun yang optimal.
13) *b(x)* = Σ_j^T *wI:* menentukan pembelajaran dasar *b(x)* sebagai jumlah dari berat daun w untuk setiap daun yang memenuhi kondisi.
14) *fk(x) = fk-1(x) + b(x):* menambahkan pohon baru ke model.
15) *f*(*x*) = Σ_k^M *fk*(*x*): model akhir f(x) adalah jumlah dari semua model pohon yang dibangun selama iterasi.

2.3.4 Extra Tree Classifier

*Extra tree classifier*s merupakan teknik *ensemble* yang mengintegrasikan banyak pohon keputusan [30]. Setiap pohon akan dibangun dengan pemilihan fitur dan titik pemisahan yang sangat acak, tujuannya meningkatkan keakuratan dan *robustness* model terhadap data. Berbeda dengan *Extra trees classifier*s, *Extra tree classifier* memiliki proses pembangunan satu pohon keputusan dengan prinsip yang sama namun tidak dalam bentuk *ensemble*. Kinerja *Extra tree classifier* lebih cepat untuk melakukan proses pelatihan model. *Extra tree classifier* banyak digunakan ketika dibutuhkan eksplorasi data awal atau ketika kekuatan komputasi yang rendah. Parameter kedua teknik tersebut sama dengan kedalaman maksimum pohon *(max_depth),* jumlah minimum sampel untuk memisahkan *node (min_samples_split),* dan jumlah fitur yang dipertimbangkan pada setiap pemisahan *(max_features).* Penyesuaian untuk parameter yang digunakan disesuaikan dengan kekhususan dan ukuran *dataset*. Tabel 2.4 menunjukkan *pseudocode* dari *Extra tree classifier*s yang isinya adalah cara kerja dari algoritma [30].

Tabel 2. 4 *Extra-Trees splitting algorithm (for numerical attributes)*

Split a node(S)

Input: the local learning subset S corresponding to the node we want to split
Output: a split [a < a_c] or nothing
– If Stop split(S) is TRUE then return nothing.
– Otherwise select K attributes {a_1,..., a_K } among all non constant (in S) candidate attributes;
– Draw K splits {s1,...,sK }, where s_i = Pick a random split(S, a_i), ∀i = 1,..., K;
– Return a split s∗ such that Score(s∗, S) = max$_{i=1,...,K}$ Score(s_i, S).
Pick a random split(S,ai)
Inputs: a subset S and an attribute a
Output: a split
– Let a^s_{max} and a^s_{min} denote the maximal and minimal value of a in S;
– Draw a random cut-point ac uniformly in [a^s_{min}, a^s_{max}];
– Return the split [a < a_c].
Stop split(S)
Input: a subset S
Output: a boolean
– If |S| < n_{min}, then return TRUE;
– If all attributes are constant in S, then return TRUE;
– If the output is constant in S, then return TRUE;
– Otherwise, return FALSE.

Sumber: Geurts (2006)

Berikut merupakan penjelasan simbol yang digunakan pada *pseudocode Extra Tree Classifier*:

1) *S*: *Subset* lokal dari data yang sesuai dengan *node* yang ingin dipecah.
2) a: atribut atau fitur yang digunakan untuk membuat *split* dalam *subest S*.
3) ac: titik potong untuk atribut a yang menentukan *split*.
4) *Stop split*(S): fungsi yang memeriksa kondisi untuk menghentikan pemisahan *node*.
5) K: jumlah atribut yang dipilih dari semua kandidat atribut yang tidak konstan dalam *subset S* untuk dipertimbangkan dalam pembuatan *split.*
6) *{a1,..., aK }:* kumpulan dari K atribut yang dipilih untuk dipertimbangkan dalam pembuatan *split.*
7) *{s1,...,sK }: kumpulan dari K split* yang dihasilkan.

8) *Pick a random split(S,ai)* : fungsi yang memilih *split* secara acak untuk *subset S* dan atribut ai.
9) a^s_{max}: nilai maksimum dari atribut a dalam *subset S.*
10) a^s_{min} *:* nilai minimum dari atribut a dalam *subset S.*
11) *Score(s, S):* fungsi yang menghitung skor dari *split S* berdasarkan *subset S.*
12) $s*$*: split* terbaik yang dipilih dari kumpulan *K split* berdasarkan skor tertinggi.
13) n_{min}*:* batas minimum jumlah dalam *subset S.* Jika jumlah sampel dalam *s* kurang dari n_{min} maka *split* dihentikan.
14) $|S|$*:* jumlah sampel dalam *subset S.*

2.3.5 Particle Swarm Optimization

Particle Swarm Optimization (*PSO*) merupakan algoritma yang terinspirasi dari perilaku sosial burung dan ikan yang efektif untuk mengoptimalkan fungsi *non linear*. Algoritma memanfaatkan informasi sosial di mana partikel dalam sistem membagi informasi mengenai posisi terbaik yang mereka sehingga kelompok dapat mencari solusi yang optimal dengan lebih efisien. Metode *PSO* tidak memerlukan komputasi yang berat sehingga dapat menjadi pilihan ideal untuk masalah optimasi [31].

Implementasi *PSO* melibatkan prinsip dasar dari *Swarm Intelligence* seperti komputasi ruang, respons terhadap lingkungan, komputasi ruang, dan kemampuan adaptasi perubahan. Algoritma dapat mengalokasikan upaya pencarian yang optimal, mencari solusi yang baik, dan mampu mengeksplorasi area baru dengan pengaturan parameter yang efisien. Pengembangan *PSO* menggarisbawahi potensi untuk meningkatkan solusi teknis yang meniru interaksi sosial dan perilaku adaptif yang terjadi dialam. Tabel 2.5 merupakan hasil *pseudocode* untuk *PSO* yang diambil dari

referensi artikel *particle swarm optimization* oleh Kennedy dan Eberhart tahun 1995 [32].

Tabel 2. 5 *pseudo-code of the PSO algorithm* (Kennedy & Eberhart, 1995)

For each particle
Initialize particle
Do until maximum iterations or minimum error criteria
For each particle
Calculate Data fitness value
If the fitness value is better than Pbest
Set Pbest is better than Gbest
Set Gbest = Pbest
For each particle
Calculate particle Velocity
Use Gbest and Velocity to update Data
Report Gbest

Sumber*:* Babaee (2021)

Berikut merupakan penjelasan simbol yang digunakan pada *pseudocode Particle Swarm Optimization*:

1) Partikel: entitas atau individu dalam populasi yang bergerak melalui ruang pencarian.
2) *Pbest*: nilai terbaik (*fitness value*) yang telah dicapai oleh partikel individu sepanjang perjalanannya.
3) *Gbest*: nilai terbaik (*fitness value*) yang telah dicapai oleh partikel dalam populasi.
4) *Fitness value*: ukuran seberapa baik solusi yang diwakili oleh partikel memenuhi tujuan optimasi.
5) *Velocity*: kecepatan partikel yang menentukan arah dan seberapa jauh partikel akan bergerak dalam ruang pencarian pada setiap iterasi.
6) Data: posisi partikel dalam ruang pencarian yang mewakili solusi potensial pada iterasi tertentu.
7) *Initialize Particle*: langkah awal menetapkan posisi awal dan kecepatan setiap partikel secara acak.

8) *Maximum Iterations*: jumlah maksimal iterasi yang diizinkan untuk algoritma PSO berjalan sebelum berhenti.
9) *Minimum Error Criteria*: kriteria kesalahan minimum yang jika tercapai akan menyebabkan algoritma berhenti.
10) *Calculate Data Fitness value*: proses evaluasi nilai *fitness* berdasarkan posisi partikel saat ini.
11) *Set Pbest*: memperbarui nilai *Pbest* jika nilai *fitness* saat ini lebih baik dari nilai *fitness* sebelumnya.
12) *Set Gbest*: memperbarui nilai *Global best* jika nilai *pbest* saat ini lebih baik dari pada *Gbest* sebelumnya.
13) Calculate *Particle Velocity*: menghitung kecepatan baru partikel berdasarkan kecepatan sebelumnya.
14) *Use Gbest and Velocity to update Data*: memperbarui posisi partikel dalam ruang pencarian berdasarkan kecepatan yang telah dihitung dan *Gbest*.

2.3.6 Ant Colony Optimization

Ant Colony Optimization (*ACO*) merupakan algoritma yang terinspirasi dari perilaku semut untuk memecahkan berbagai masalah optimasi [33]. Perilaku semut sebagai serangga sosial yang tinggal dalam koloni dan menggunakan feromon untuk menemukan rute saat berjalan. Semut cenderung mengikuti jalur memalui kepadatan feromon tinggi yang akan memudar seiring waktu untuk mengurangi intensitas jalur yang kurang populer sehingga semut akan berkonvergensi cepat ke solusi *sub-optimal*. Algoritma *ACO* merepresentasikan masalah sebagai graf dengan nodus dan tepian dengan pembaruan feromon dan aturan transisi probabilistik untuk membuat keputusan. *ACO* memiliki beberapa keunggulan di antara lain komputasi terdistribusi, memberikan umpan balik positif, fleksibilitas dalam aplikasi yang dinamis, memungkinkan penemuan solusi optimal dan

rute ulang yang dinamis ketika terjadi kerusakan pada nodus. Tabel 2.6 merupakan *pseudocode* untuk *ACO* [33].

Tabel 2. 6 *Pseudocode Ant Colony Optimization*

1.	*Initialize:*
	Set t := 0 {t is the time counter}
	Set NC := 0
	For every edge (i,j) set an initial value T_{ij} *(t)= c for trail intensity and* $\Delta_{T_{ij}} = 0$
	Place the m ants on the n nodes
2.	*Set s := 1 { s is the tabu list index}*
	For k := 1 to m do
	Place the starting town of the kth ant in $tabu_k(s)$
	Repeat until tabu list is full
	{this step will be repeated (n - 1) times}
	Set s := s + 1
	For k := 1 to m do
	Choose the town j to move to, with probability
	$P_{ij}^k(t)$ *given by Eq. (4)*
	{at time t the kth ant is on town i = $tabu_k(s - 1)$*}*
	Move the kth ant to the town j
	Insert town j in $tabu_k(s)$
4.	*For k := 1 to m do*
	Move the kth ant from $tabu_k(n)$ *to* $tabu_k(1)$
	Compute the length L_k*, of the tour described by the kth ant*
	Update the shortest tour found
	For every edge (i,j)
	For k:= 1 to m do
	$\Delta T_{ij}^k = \begin{cases} \frac{Q}{L_k} & \text{if (i,j) E tour described by } tabu_k \\ 0 & \text{otherwise} \end{cases}$
	$\Delta_{T_{ij}} := \Delta_{T_{ij}} + \Delta T_{ij}^k$;
5.	*For every edge (i, j) compute* T_{ij} *(t + n)*
	according to equation $T_{ij} + (t + n) = p \cdot T_{ij}(t) + \Delta_{T_{ij}} +$
	Set t:=t+n
	Set NC := NC + 1
	For every edge (i, j) set $\Delta_{T_{ij}} := 0$

6.	*If (NC < NCMAX) and (not stagnation behavior)*
	then
	Empty all tabu lists
	Goto step 2
	else
	Print shortest tour
	stop

Sumber: Dorigo (1996)

Berikut merupakan penjelasan simbol yang digunakan pada *pseudocode Ant Colony Optimization*:

1) T_{ij}: intensitas jejak feromon pada tepi antara *node* i dan j.
2) t: penghitung waktu atau iterasi.
3) NC: penghitung siklus atau jumlah iterasi.
4) c: nilai awal untuk intensitas jejak feromon.
5) $\Delta_{T_{ij}}$: perubahan dalam intensitas jejak feromon pada tepi antara *node* i dan j.
6) m: jumlah semut dalam populasi.
7) n: jumlah *node* atau kota dalam masalah.
8) s: indeks daftar tabu yang melacak jalur yang diambil oleh setiap semut.
9) tabuk(s): daftar tabu yang mencatat kota ke-s yang dikunjungi oleh semut ke-k.
10) $P_{ij}^{k}(t)$*:* probabilitas bahwa semut ke-k memilih untuk bergerak ke kota j dari kota i pada waktu t.
11) L_k: panjang tur yang dijelaskan oleh semut ke-k.
12) Q: konstanta yang digunakan dalam perhitungan perubahan intensitas feromon.
13) NCMAX: jumlah maksimum siklus atau iterasi yang diizinkan.
14) p: faktor penguapan feromon yang mengontrol laju penguapan jejak feromon.

15) *(i,j) E tour described by tabuk*: kondisi yang memeriksa apakah tepi (i,j) bagian dari tur yang dijelaskan oleh daftar tabu.

2.4 Tools

2.4.1 Python

Python adalah bahasa pemrograman tingkat tinggi, interaktif, dan berorientasi objek yang kuat. *Python* diciptakan oleh Guido Van Rossum pada tahun 1980-an. Bahasa pemrograman *python* berorientasi kepada objek, artinya bahasa *python* dapat memodelkan entitas dunia nyata. Salah satu kelebihan *python* adalah kodenya yang dikompilasi baris per-baris yang membuat lebih mudah untuk melakukan *debugging* [34]. *Python* merupakan bahasa pemrograman populer ini karena *python* merupakan bahasa pemrograman yang mudah, kurva belajar cepat dan memiliki banyak paket berkualitas tinggi.

2.4.2 Jupyter notebook

Jupyter notebook adalah *platform* yang memungkinkan pengguna untuk dapat membuat dokumen interaktif yang menggabungkan kode sumber dan *output* dari eksekusi kode. *Jupyter notebook* merupakan platform yang digemari oleh para peneliti karena kemampuannya yang terstruktur dari analisis data dan model yang dibangun. *Platform jupyter notebook* biasa digunakan untuk pengembangan komputasi yang ditulis oleh *python* [35].