BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian terdahulu dilakukan untuk menyediakan landasan teoritis yang membantu dalam mengidentifikasi celah pada penelitian yang ada, serta memperkuat validitas dan relevansi dari penelitian saat ini. Penelitian ini berfokus pada penerapan *hyperparameter optimization* pada klasifikasi *Collection Intensity Scoring* dan prediksi *channel recommendation*. Sebagai acuan, berikut terdapat penelitian terdahulu yang serupa yang disajikan dalam Tabel 2.1.

Tabel 2.1 Penelitian Terdahulu

Nama Penulis	Nama Artikel	Nama Jurnal	Metode	Hasil
Y. Li, W. Chen [18]	*A Comparative Performance Assessment of Ensemble Learning for Credit Scoring*	*Mathematics* (2020)	*RF*, *AdaBoost*, *XGBoost*, *LightGBM*, *Stacking*, *ANN*, *LR*, *DT*, *SVM*, dan *NB* dengan metode *Grid Search*	*RF* dengan metode *Grid Search* mencapai nilai *accuracy* sebesar 81.05%.
J. Xu, Z. Lu, Y. Xie [17]	*Loan default prediction of Chinese P2P market: a machine learning methodology*	*Scientific Reports* (2021)	*RF*, *XGBT*, *GBM*, *NN*	*RF* mengungguli model lainnya dengan nilai akurasi sebesar 98.4%.
S. K. Trivedi [16]	*A study on credit scoring modeling with different feature selection and machine learning approaches*	*Technology in Society* (2020)	*Bayesian*, *Naïve Bayes*, *SVM*, *DT*, *RF* dengan metode *feature selection*	*RF* memperoleh nilai *accuracy* terbaik sebesar 93%.
K. K. Jena, S. K. Bhoi, T. K. Malik, K. S. Sahoo, N. Z. Jhanjhi, S. Bhatia, F. Amsaad [20]	*E-Learning Course Recommender System Using Collaborative Filtering Models*	*Electronics* (2023)	*KNN*, *SVD*, *SVF*	*KNN* memperoleh nilai *MAE* sebesar 0.0142.

Nama Penulis	Nama Artikel	Nama Jurnal	Metode	Hasil
S. G. K. Patro, B. K. Mishra, S. K. Panda, R. Kumar, H. V. Long, D. Taniar, I. Priyadarshini [19]	*A Hybrid Action-Related K-Nearest Neighbour (HAR-KNN) Approach for Recommendation Systems*	*IEEE Access* (2020)	*Hybrid Action-Related K-Nearest Neighbour* (*HAR-KNN*) dengan *KNN,*	*HAR-KNN* memperoleh nilai *MAE* sebesar 0.7165.
S. Jaggia, A. Kelly, K. Lertwachara, L. Chen [25]	*Applying the CRISP-DM Framework for Teaching Business Analytics*	*Decision Sciences Journal of Innovative Education* (2s020)	*Cross Industry Standard Process-Data Mining* (*CRISP-DM*) *Framework*	Penerapan *CRISP-DM Framework* terbukti berhasil dalam mendukung peningkatkan pemahaman dan pelaksanaan setiap tahap dalam proyek analitik.
W. S. Fana, R. Soviab, R. Permana, M. A. Islam [26]	*Data Warehouse Design with ETL Method (Extract, Transform, And Load) for Company Information*	*International Journal of Artificial Intelligence Research* (2021)	*Extract, Transformation, Load* (*ETL*)	Penerapan *Extract, Transform, Load* (*ETL*) pada implementasi *data warehouse* berhasil membantu penggabungan hasil ekstrasi dari sumber data lain, melakukan transformasi menjadi data yang sesuai dengan pengambilan keputusan.
T. Wang, X. Wang, R. Ma, X. Li, X. Hu, F. T. S. Chan, J. Ruan [22]	*Random Forest-Bayesian Optimization for Product Quality Prediction with Large Scale Dimensions in Process Industrial Cyber-Physical Systems*	*IEEE Internet of Things Journal* (2020)	*LR, DT, Support Vector Classifier* (*SVC*), *Background Propagation Neural Network* (*BPNN*), *RF, Random Forest - Bayesian Optimization* (*RF_BO*)	*Random Forest* dengan metode *Bayesian Optimization* (*RF_BO*) menghasilkan nilai *accuracy* sebesar 90.33%.
M. Y. Shams, A. M. El-kenawy, A. Ibrahim, F. M. Talaat, Z. Tarek [21]	*Water Quality Prediction Using Machine Learning Models based on Grid Search Method*	*Multimedia Tools and Applications* (2023)	*KNN, DT, SVR, MLP* dengan metode *Grid Search*	*KNN Regressor* dengan metode *Grid Search* mencapai nilai *MAE* sebesar 0.0009.

Nama Penulis	Nama Artikel	Nama Jurnal	Metode	Hasil
S. Lee, J. H. Bae, J. Hong, D. Yang, P. Panagos, P. Borelli, J. E. Yang, J. Kim, K. J. Lim [23]	*Estimation of rainfall erosivity factor in Italy and Switzerland using Bayesian optimization based machine learning models*	Catena (2022)	*DT, KNN, RF, GB, XGBoost* dengan metode *Bayesian Optimization*	*KNN* dengan metode *Bayesian Optimization* menghasilkan nilai *MAE* sebesar 20.576 MJ mm ha^{-1} h^{-1} (erositivitas curah hujan)

Tabel 2.1 menunjukkan penelitian terdahulu dalam melakukan pembuatan model klasifikasi *credit scoring* dan prediksi *recommendation system* dengan penerapan *hyperparameter optimization*. Penelitian ini menggunakan metode pada penelitian terdahulu. Penelitian ini menggunakan *Cross Industry Standard Process-Data Mining* (*CRISP-DM*) *framework* seperti yang digunakan pada [25] karena mencakup pemahaman dalam konteks bisnis dengan tahapan yang terstruktur dengan proses perubahan data menjadi wawasan dan strategi bisnis yang dapat ditindaklanjuti. Selain itu, penelitian ini menggunakan proses *Extract, Transform, Load* (*ETL*) pada [26] karena kemampuan dalam mengumpulkan dan mengolah data dari berbagai sumber menjadi data yang terintegrasi untuk pengambilan keputusan bisnis. Belum terdapat penelitian yang dilakukan terkait penggunaan *CRISP-DM framework* dan *ETL* untuk klasifikasi *Collection Intensity Scoring* (*CIS*) dan prediksi *channel recommendation*.

Dalam klasifikasi *credit scoring*, terdapat salah satu algoritma *machine learning* yang paling populer untuk digunakan, yaitu *Random Forest* karena menghasilkan nilai akurasi tertinggi. Walaupun memiliki performa yang bagus, *Random Forest* dapat menjadi kompleks ketika memiliki fitur dan observasi yang banyak sehingga berdampak pada waktu pelatihan. Terdapat penerapan *Random Forest* pada [16] dengan akurasi 93.12% memiliki waktu pelatihan selama 16.20 detik. Model *Random Forest* pada [17] memperoleh nilai akurasi 98.4% dan tidak menjelaskan secara eksplisit terkait waktu pelatihan, namun terdapat 58.477 observasi dan 28 fitur sehingga menyebabkan kompleksitas model yang lebih tinggi. Oleh karena itu, pengoptimalan waktu pelatihan dari model *Random Forest* dibutuhkan tanpa mengorbankan tingkat akurasi. Terdapat sebuah penelitian yang menerapkan

Random Forest dengan metode optimasi *Grid Search* menghasilkan akurasi tertinggi sebesar 81.05% dengan waktu pelatihan 0.8949 detik [18]. Akan tetapi, penelitian tersebut terbatas pada satu metode optimasi sehingga ada kemungkinan bahwa model yang dihasilkan tidak sepenuhnya dioptimalkan. Di sisi lain, terdapat *Random Forest* dengan metode *Bayesian Optimization* pada [22] dengan akurasi 90.33% dan waktu pelatihan 4.348 detik, namun hanya terbatas pada cakupan industri *cyber-physical* dan belum diterapkan pada *Collection Intensity Scoring* (*CIS*).

Selain klasifikasi *credit scoring*, terdapat algoritma *machine learning* yang sering digunakan untuk prediksi *recommendation system*, yaitu *K-Nearest Neighbors* karena menghasilkan nilai *Mean Absolute Error* (*MAE*) terkecil. Namun, pendekatan ini memiliki performa yang bervariasi tergantung pada nilai k tetangga yang dipilih. Sebuah penelitian yang menggunakan *K-Nearest Neighbors* dengan nilai *MAE* 0.7165 pada nilai *k*=10, sedangkan terdapat nilai k 50 dengan *MAE* 0.8245 [19]. Selain itu, penerapan *K-Nearest Neighbors* pada [20] dengan nilai *MAE* 0.0142 tidak menyebutkan nilai k, melainkan menggunakan *training-testing ratio* dari 60:40 hingga 80:20. Oleh sebab itu, pemilihan jumlah k tetangga yang optimal dibutuhkan untuk mencapai performa yang baik dalam *K-Nearest Neighbors*. Terdapat penerapan *K-Nearest Neighbors Regressor* pada [21] dengan *Grid Search* menghasilkan nilai *MAE* 0.0009 pada nilai *k*=1, namun jumlah tetangga yang digunakan memiliki rentang yang luas yang berdampak pada waktu komputasi. Di sisi lain, *K-Nearest Neighbors* pada *Bayesian Optimization* memiliki *MAE* 20.576 satuan erosivitas curah hujan (MJ mm ha^{-1} h^{-1}), dengan R^2 sebesar 0.864 pada nilai *k*=10, namun memerlukan waktu selama 264 detik [23]. Belum terdapat penelitian yang dilakukan terkait penerapan *Grid Search* dan *Bayesian Optimization* pada model *K-Nearest Neighbors Regressor* untuk *channel recommendation*.

Penelitian ini memiliki kebaruan berupa klasifikasi intensitas penagihan nasabah dalam *Collection Intensity Scoring* (*CIS*) dan prediksi jumlah interaksi yang akan direspon nasabah dalam *channel recommendation* dimana belum terdapat penelitian terkait hal tersebut. Dengan demikian, penelitian ini

membandingkan metode optimasi *Grid Search* dan *Bayesian Optimization* terhadap penggunaan model klasifikasi *Random Forest* dan model prediksi *K-Nearest Neighbors Regressor*.

2.2 Tinjauan Teori

2.2.1 *Financial Technology* (*Fintech*)

Financial Technology (*Fintech*) merupakan perusahaan yang memadukan layanan keuangan dengan teknologi inovatif yang ditawarkan kepada penyedia layanan keuangan [27]. *Fintech* memiliki istilah sebagai sektor dalam suatu bisnis yang mampu memberikan inovasi dalam bidang layanan keuangan dengan memanfaatkan perkembangan teknologi. *Fintech* hadir untuk memberikan perubahan dan membentuk kembali industri keuangan dengan menghemat biaya, meningkatkan kualitas layanan keuangan, dan menciptakan lanskap keuangan yang lebih beragam dan kuat [27]. *Fintech* memiliki berbagai inovasi yang memungkinkan akses menuju layanan keuangan melalui perangkat seluler bagi banyak orang yang tidak memiliki rekening bank di dunia [28]. *Fintech* tidak hanya terbatas pada transaksi pembayaran, tetapi juga mencakup berbagai jenis layanan berupa investasi, pinjaman *P2P*, *crowdfunding*, dan agregator pasar [29].

2.2.1.1 *Peer-to-Peer Lending (P2P Lending)*

Peer-to-Peer Lending (*P2P Lending*) memiliki definisi sebagai sebuah *platform* pasar berbasis elektronik dimana pemberi pinjaman individual menyediakan pinjaman kepada peminjam individual [30]. Dengan kata lain, *P2P lending* merupakan salah satu bidang dari *fintech* yang ditujukan untuk kebutuhan peminjaman *online* yang disediakan oleh pemberi pinjaman terhadap para peminjam tanpa melibatkan perantara atau pihak kedua. Dengan memanfaatkan perkembangan teknologi seperti *cloud computing*, *big data*, dan jaringan sosial, *P2P* menerapkan pembagian informasi dan pencarian untuk memfasilitasi *screening* untuk peminjam. *Platform* tersebut menghilangkan biaya *overhead* yang tinggi dari bank tradisional dan memungkinkan pengurangan jumlah transaksi [31].

2.2.1.2 *Credit Scoring*

Credit scoring merupakan proses penilaian yang penting dari sistem *credit risk management* dalam suatu lembaga keuangan untuk melakukan prediksi pada risiko dalam aplikasi pinjaman [32]. Adapun pertimbangan yang dilakukan di balik *credit scoring* adalah menghasilkan suatu nilai yang mampu memberikan perbedaan antara para pemohon pinjaman menjadi seseorang yang layak kredit dan sangat mungkin untuk membayar pinjamannya, serta seseorang yang berisiko tidak mungkin untuk melakukan pembayaran pada pinjamannya [33]. Hal ini penting bagi perusahaan untuk mengumpulkan informasi nasabah untuk mengelola risiko keuangannya dan membawa suatu hasil keputusan yang penting untuk memberikan pinjaman atau tidak [34].

Credit scoring memiliki metode yang dapat ditetapkan dari lembaga keuangan sendiri atau menggunakan layanan pihak ketiga, yaitu sistem *Fair Isaac Corporation* (*FICO*) [35]. Adapun beberapa parameter dari *FICO* untuk menghitung skor kredit seorang konsumer dan mengevaluasi kelayakan kredit mereka, seperti *payment history*, *amounts owed*, *length of credit history*, *new credit* sebesar, dan *credit mix*. FICO mempertimbangkan apakah pembayaran kredit telah dilakukan tepat waktu, rasio penggunaan kredit konsumer pada semua kredit yang tersedia, panjangnya riwayat kredit, dan variasi jenis kredit yang digunakan [9]. Selain *FICO*, terdapat beberapa perusahaan menetapkan metode *credit scoring* sesuai dengan kebutuhan spesifik tersendiri. Dalam hal ini, perusahaan *P2P Lending* menggunakan *Collection Intensity Scoring* (*CIS*) yang berfokus pada penilaian intensitas penagihan nasabah. Adapun beberapa faktor yang digunakan dalam *CIS*, seperti *communication channel interaction*, *average interaction to payment*, dan *paid loans*. *CIS* menekankann pentingnya komunikasi dalam mencapai responsitivitas yang cepat dan kemampuan dalam melakukan pembayaran kembali pinjaman.

2.2.2 *Recommendation System*

Recommendation system merupakan *software* sekaligus teknik yang menawarkan rekomendasi untuk pengguna [11]. *Recommendation system* dikategorikan sebagai salah satu bagian dari sistem penyaringan informasi yang dirancang untuk memprediksi preferensi yang akan diberikan pengguna terhadap suatu topik [36]. *Recommendation system* bertujuan untuk menyediakan daftar rekomendasi *item* yang ditawarkan oleh sebuah layanan [37]. *Recommendation system* menyediakan rekomendasi dengan menjaga minat pengguna dan memanfaatkan informasi kontekstual ke dalam akun [11]. *Recommendation system* di perusahaan *modern* banyak digerakkan oleh data dan mengandalkan aspek kognitif pengguna, seperti kepribadian, perilaku, dan sikap [38].

Recommendation system dikategorikan menjadi 3 jenis berdasarkan tekniknya, yaitu *content-based filtering*, *collaborative filtering*, dan *hybrid filtering*. *Content-based filtering* memberikan rekomendasi elemen bagi pengguna yang secara praktis identik dengan elemen sebelumnya yang telah dipilih atau diinginkan oleh pengguna. *Collaborative filtering* mengklasifikasikan kesukaan pelanggan dan merekomendasikan apa yang disukai oleh kebanyakan pelanggan. Jenis *filtering* ini merekomendasikan produk dengan preferensi yang sebelumnya identik dengan pengguna aktif atau pengguna target. *Hybrid filtering* merupakan kombinasi dari *content-based* dan *collaborative filtering* yang berarti klasifikasi pelanggan dan pemberian rekomendasi dengan menggunakan kedua metode *filtering* [39].

2.2.3 *Extract, Transformation, Load* (*ETL*)

Proses *ETL* merupakan urutan operasi yang teratur berupa *Extract*, *Transformation*, dan *Load* dengan tujuan untuk melakukan pemrosesan data sumber dengan sistematis sehingga dapat tersedia dalam format yang lebih nyaman untuk penggunaan yang diinginkan [40]. Tujuan dari proses ini adalah untuk menggabungkan data yang tersebar, berantakan, dan tidak konsisten pada suatu perusahaan dan menyediakan dasar analisis untuk pengambilan keputusan perusahaan [41]. Data yang digunakan dalam proses *ETL* dapat berasal dari

berbagai sumber, termasuk *Enterprise Resource Planning* (*ERP*), *file* biasa, dan *spreadsheet* [26].

Extract merupakan tahap pertama dalam proses memasukkan data ke dalam lingkungan *data warehouse*. *Extract* memiliki makna membaca dan memahami sumber data dan menyalin data yang dibutuhkan ke dalam proses *ETL* untuk manipulasi lebih lanjut. Setelah data dilakukan *extract* dalam proses *ETL*, terdapat banyak *transformation* yang potensial, seperti *data cleansing* (mengoreksi kesalahan ejaan, menyelesaikan konflik *domain*, menangani elemen yang hilang, atau mengurai ke dalam format standar), *data combination* dari berbagai sumber, dan *data de-duplication*. Proses *ETL* menambahkan nilai pada data dengan langkah-langkah pembersihan dan penyesuaian data dengan mengubah data dan menyempurnakannya [42]. Pada penelitian ini, *data filtering* dan *data aggregation* sebagai bagian dari *data cleansing* digunakan untuk memberikan penyempurnaan pada data yang digunakan. Data kemudian dilakukan *load* ke dalam *database* target atau destinasi apapun [43].

2.2.4 *Grid Search*

Grid search merupakan sebuah metode *hyperparameter optimization* yang khas dan melibatkan pencarian menyeluruh dari ruang pencarian yang telah ditentukan sebelumnya [18]. *Grid Search* berupaya untuk mencari secara ekstensif melalui semua kombinasi *hyperparameter* yang berpotensi dalam suatu rentang atau kumpulan nilai tertentu. Hal ini dilakukan dengan membuat *grid* terlebih dahulu dari semua kemungkinan kombinasi *hyperparameter*, dan melatih dan menguji model pada suatu validasi atau *cross-validation* untuk setiap kombinasi [21]. *Grid search* tidak terbatas pada satu model saja, tetapi dapat diterapkan di seluruh *machine learning* sehingga parameter terbaik dapat diidentifikasi untuk model tersebut [44].

2.2.5 *Bayesian Optimization*

Bayesian Optimization merupakan metode yang digunakan untuk menemukan *extrema* dari fungsi *black-box* [45]. *Bayesian Optimization* bermanfaat dalam menemukan *hyperparameter* yang optimal untuk sebuah

model *machine learning*. *Bayesian Optimization* bekerja dengan baik ketika *dataset* untuk klasifikasi bersifat *non-linear*, kompleks, dan *noisy* karena komputasi untuk mengidentifikasi *hyperparameter* yang mahal sehingga dapat berpengaruh pada performa model [46]. Proses *Bayesian Optimization* mencakup dua langkah utama, yaitu 1) memperkirakan fungsi *black-box* dari data melalui *probabilistic surrogate model* dengan *Gaussian Process* (*GP*) yang dikenal sebagai *response surface*, 2) memaksimalkan sebuah *acquisition function* untuk *trade-off* antara eksplorasi dan eksploitasi berdasarkan ketidakpastian dan optimalitas dari *response surface* [47]. *Bayesian Optimization* mencakup informasi sebelumnya tentang fungsi f dan memperbarui informasi posterior yang membantu mengurangi *loss* dan memaksimalkan akurasi model [46]. Berikut terdapat penerapan *Gaussian Process* dari *Bayesian Optimization* yang ditunjukkan sebagai berikut [48].

$$f(x) \sim GP\big(m(x), k(x, x')\big) \tag{2.1}$$

Rumus 2.1 *Prior Distribution*

Persamaan (2.1) menunjukkan persamaan *Gaussian Process* yang mengambil *prior distribution* sebagai distribusi normal multivariat dengan *mean vector* dan *covariance matrix* tertentu. Persamaan ini mencakup *x* sebagai titik dalam *input space*, $m(x)$ sebagai *mean function* untuk *mean vector*, $k(x, x')$ sebagai *covariance function* atau *kernel* pada setiap pasangan titik x dan x' untuk *covariance matrix*. *Kernel* ini dipilih sehingga titik x dan x' memiliki korelasi positif yang besar, menandakan bahwa mereka harus memiliki nilai fungsi yang lebih mirip dibandingkan daripada titik yang berjauhan. Selanjutnya, dilakukan penghitungan *posterior probability function* sebagai *conditional distribution* untuk mengambil nilai dari $f(x)$ dan memprediksinya pada beberapa titik baru x_*. Penghitungan *posterior probability distribution* ini mencakup *posterior mean* dan *posterior variance* yang dilakukan melalui persamaan (2.2), (2.3), (2.4) secara berurutan [48].

$$f(x_*) \mid f(X) \sim GP\left(\mu_{post}(x_*), \sigma^2_{post}(x_*)\right) \tag{2.2}$$

Rumus 2.2 *Posterior Probability Distribution*

$$\mu_{post}(x_*) = k(x_*, X)[k(X, X) + \sigma_n^2 I]^{-1} Y \quad (2.3)$$

Rumus 2.3 *Posterior Mean*

$$\sigma^2_{post}(x_*) = k(x_*, x_*) - k(x_*, X)[k(X, X) + \sigma_n^2 I]^{-1} k(X, x_*) \quad (2.4)$$

Rumus 2.4 *Posterior Variance*

Persamaan (2.2) menampilkan persamaan *posterior probability distribution*, dimana $\mu_{post}(x_*)$ adalah *posterior mean*, $\sigma^2_{post}(x_*)$ adalah *posterior variance*. *Posterior mean* dalam persamaan (2.3) merupakan *weighted average* antara *prior mean* $m(x)$ dengan prediksi berdasarkan data yang diobservasi $f(X)$ dengan bobot yang bergantung pada *kernel*. Persamaan ini mencakup $k(x_*, X)$ sebagai *covariance* antara titik prediksi dengan titik yang diobservasi, $k(X, X)$ sebagai *covariance* antara titik yang diobservasi, dan $\sigma_n^2 I$ sebagai variansi *noise* yang disesuaikan dengan *covariance* $k(X, X)$, dan Y sebagai nilai-nilai target yang diobservasi. *Posterior variance* dalam persamaan (2.4) dihitung dengan mengambil *prior covariance* $k(x, x')$ dan menyesuaikannya dengan mengurangi *variance* dari data yang diobservasi $f(X)$.

Penerapan *acquisition function* kemudian dilakukan dengan *expected improvement* untuk mengambil nilai yang diharapkan dari peningkatan dan memilih *x* untuk memaksimalkannya [48]. Hasil penerapan ini akan digunakan untuk memperbarui model statistikal *Gaussian Process* [46]. Penghitungan *expected improvement* dapat dilakukan melalui persamaan (2.5) [48].

$$EI(x) = \mathbb{E}\left[max(f(x) - f(x^+), 0)\right] \quad (2.5)$$

Rumus 2.5 *Expected Improvement*

Persamaan (2.5) menampilkan persamaan *expected improvement*, dimana $\mathbb{E}$ adalah nilai yang diharapkan yang diambil dari *posterior distribution* dari evaluasi *f*, dan x^+ adalah titik dengan nilai terbaik yang diamati [48]. $EI(x)$ bernilai positif ketika nilai yang diprediksi lebih tinggi dibandingkan nilai terbaik sejauh ini. Selain itu, $EI(x)$ ditetapkan menjadi nilai 0 [22].

2.2.6 *Evaluation Metrics*

Penelitian ini melibatkan *evaluation metrics* dengan tujuan menilai performa dari model klasifikasi [16], [17], [18]. Dalam hal ini, terdapat penghitungan *evaluation metrics* oleh *confusion matrix* yang menunjukkan 4 komponen, yaitu *True Positive* (*TP*), *False Positive* (*FP*), *False Negative* (*FN*), dan *True Negative* (TN). Berikut terdapat *confusion matrix* yang ditunjukkan pada Gambar 2.1.

		Predicted	
		Positives	Negatives
Actual	Positives	TP	FN
	Negatives	FP	TN

Gambar 2.1 *Confusion Matrix*
Sumber: [49]

Berdasarkan *confusion matrix* dari Gambar 2.1, terdapat *False Positive Rate* (*FP Rate*) atau yang dikenal sebagai tingkat misklasifikasi dari nilai positif dianggap baik jika nilainya rendah atau mendekati nol. *False Negative Rate* (*FN Rate*) digunakan untuk nilai negatif yang salah diklasifikasikan [16]. Hal ini sebaliknya dengan *True Positive* dan *True Negative* yang merupakan tingkat klasifikasi dari nilai positif yang bernilai positif dan nilai positif yang salah diklasifikasikan secara berturut-turut. Berdasarkan penghitungan yang dilakukan *confusion matrix*, maka dapat dihasilkan suatu turunan dari *evaluation metrics* berupa *accuracy*, *precision*, *recall*, dan *f*1-*score*.

2.2.6.1 *Accuracy*

Accuracy adalah performa dari setiap model klasifikasi dihitung dengan *confusion matrix*. *Accuracy* merupakan rasio dari semua data yang diklasifikasikan secara akurat. Semakin tinggi nilai akurasi yang didapatkan dari suatu model, maka hal ini mampu menghasilkan performa model yang baik. *Accuracy* dapat diperoleh dalam persamaan (2.6), yaitu sebagai berikut [16].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.6)$$

Rumus 2.6 *Accuracy*

Dimana:

- TP adalah *True Positive*.
- TN adalah *True Negative*.
- FP adalah *False Positive*.
- FN adalah *False Negative*.

2.2.6.2 *Precision*

Precision menunjukkan berapa banyak sampel yang diprediksi positif atau *true positive*. *Precision* terdiri atas *TP* sebagai *True Positive* dan *FP* sebagai *False Positive*. *Precision* dapat diperoleh melalui persamaan (2.7), yaitu sebagai berikut [17].

$$Precision = \frac{TP}{TP + FP} \tag{2.7}$$

Rumus 2.7 *Precision*

2.2.6.3 *Recall*

Recall atau *sensitivity* di klasifikasi biner menunjukkan berapa banyak nilai positif dalam sampel yang diprediksi dengan benar. *Recall* terdiri atas *TP* sebagai *True Positive* dan *FN* sebagai *False Negative*. *Recall* dapat diperoleh melalui persamaan (2.8), yaitu sebagai berikut [17].

$$Recall = \frac{TP}{TP + FN} \tag{2.8}$$

Rumus 2.8 *Recall*

2.2.6.4 *F1-Score*

F1-Score merupakan rata-rata harmonik dari *precision* dan *recall* [17]. Dengan kata lain, *f1-score* merupakan salah satu cara yang efektif untuk menghitung akurasi dari metode klasifikasi dengan menggunakan nilai rata-rata (*mean*) dari *precision* dan *recall* dalam menghitung nilainya [16]. *F1-Score* dapat dihitung dengan persamaan (2.9), yaitu sebagai berikut [17].

$$F1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \quad (2.9)$$

Rumus 2.9 *F1-Score*

Penelitian ini juga melibatkan penggunaan *evaluation metrics* untuk penilaian performa model regresi atau prediksi [19], [20]. *Evaluation metrics* yang akan digunakan mencakup *Mean Absolute Error* (*MAE*), *Mean Square Error* (*MSE*), dan *Root Mean Square Error* (*RMSE*). Adapun penjelasan dari setiap *evaluation metrics* dalam penilaian model regresi, yaitu sebagai berikut.

2.2.6.5 *Mean Absolute Error*

Mean Absolute Error (*MAE*) berfungsi untuk mengevaluasi ukuran antara nilai aktual dan nilai prediksi. Hal ini menunjukkan perbedaan antara nilai target dengan nilai prediksi [50]. *MAE* berkisar dari nol hingga tak terbatas, dimana semakin rendah nilai mendefinisikan akurasi yang lebih baik dan tak terhingga merupakan kesalahan maksimum pada peringkat prediksi [19]. Terdapat penghitungan *Mean Absolute Error* yang ditampilkan dalam persamaan (2.10) [50].

$$MAE = \frac{\sum_{i=1}^{n} |X_i - \hat{X}_i}{n} \quad (2.10)$$

Rumus 2.10 *Mean Absolute Error*

Dimana:

- $\hat{X}_i$ adalah nilai yang diprediksi
- X_i adalah nilai yang sebenarnya.
- n adalah jumlah observasi.

2.2.6.6 *Mean Square Error*

Mean Square Error (*MSE)* mengambil kuadrat dari perbedaan antara nilai aktual dan nilai yang dievaluasi. Pengambilan kuadrat tersebut dilakukan untuk menurunkan nilai negatif. Penghitungan *Mean Square Error* dapat dilakukan melalui persamaan (2.11) [50].

$$MSE = \frac{\sum_{i=1}^{n}\left(X_i - \hat{X}_i\right)^2}{n} \quad (2.11)$$

Rumus 2.11 *Mean Square Error*

2.2.6.7 *Root Mean Square Error*

Root Mean Square Error (*RMSE*) mendeteksi tingkat *error* dari model regresi dan memeriksa ukuran *error* dengan ukuran nilai target [50]. Nilai *RMSE* yang lebih rendah mendefinisikan akurasi prediksi dari hasil *Recommendation System* (*RS*) yang lebih baik [19]. Terdapat penghitungan *Root Mean Square Error* yang dilakukan pada persamaan (2.12) [50].

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}\left(X_i - \hat{X}_i\right)^2}{n}} \quad (2.12)$$

Rumus 2. 12 Root Mean Square Error

2.2.7 *Statistical Tests*

Statistical tests digunakan untuk menganalisis data yang memerlukan asumsi agar hasilnya bersifat *valid* [51]. Dalam penelitian ilmiah, *statistical tests* dibuat berdasarkan data kategorikal dan data non-kategorikal [52]. Tujuan utama dari *statistical tests* adalah menilai kebenaran suatu klaim atau hipotesis tentang suatu model atau nilai *parameter* berdasarkan data yang diamati [53]. *Statistical tests* pada umumnya dilakukan menggunakan hasil eksperimen pada sejumlah *dataset* untuk membuktikan bahwa algoritma tertentu lebih unggul dibandingkan dengan algoritma *benchmark* [54]. *Statistical tests* membantu para peneliti dan praktisi untuk menarik kesimpulan tentang populasi yang diminati dengan dari sampel yang representatif [55]. Terdapat dua jenis *statistical tests* yang digunakan dalam penelitian ini, yaitu sebagai berikut.

2.2.7.1 Shapiro-Wilk *Test*

Shapiro-Wilk *Test* adalah salah satu metode yang krusial untuk menguji asumsi normalitas [56]. Shapiro-Wilk *test* digunakan untuk menguji apakah sebuah sampel acak dari seluruh populasi berasal dari populasi yang terdistribusi normal [57]. Uji normalitas ini membantu dalam menentukan penggunaaan uji *parametric* atau uji *non-parametric* dalam

menganalisis data [56]. Adapun penghitungan Shapiro-Wilk *test* yang ditunjukkan pada persamaan (2.13) [58].

$$W = \frac{\left(\sum_{i=1}^{n} a_i x_{(i)}\right)^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \quad (2.13)$$

Rumus 2.13 *Shapiro-Wilk Test*

Dimana:

- $x_{(i)}$ adalah nilai sampel yang diurutkan dari terkecil hingga terbesar.
- a_i adalah konstanta yang dihasilkan dari rata-rata, varian dan kovarian dari urutan statistik dari sampel berukuran n dari distribusi normal.
- $\bar{x}$ adalah rata-rata sampel.
- n adalah ukuran sampel.

2.2.7.2 *T-Test*

T-Test merupakan uji statistik yang digunakan untuk membandingkan rata-rata dari dua kelompok [59]. *T-test* dibagi menjadi dua jenis, yaitu *independent sample t-test* dan *paired t-test*. *Independent sample t-test* digunakan ketika dua kelompok yang dibandingkan tidak berhubungan atau sama lain. *Paired t-test* digunakan ketika dua kelompok yang dibandingkan saling bergantung satu sama lain [59]. Dalam penelitian ini, *paired t-test* digunakan untuk membuktikan adanya perbedaan kinerja yang signifikan antara model-model yang dibandingkan berdasarkan metrik *accuracy* dan *Mean Absolute Error* (*MAE*). *Paired t-test* bermanfaat dalam menganalisis sekumpulan data yang sama yang telah diukur dalam dua kondisi yang berbeda, perbedaan pengukuran pada subjek yang sama sebelum dan sesudah perlakuan, atau perbedaan antara dua perlakuan yang diberikan pada subjek yang sama [60]. Adapun penghitungan dari *paired t-test* yang dilakukan melalui persamaan-persamaan sebagai berikut [61].

$$d_i = y_i - x_{\mathrm{i}} \tag{2.14}$$

Rumus 2.14 *Pairwise Difference*

Persamaan (2.14) menunjukkan persamaan *pairwise difference* atau perbedaan berpasangan antara nilai-nilai observasi yang sesuai dari dua sampel yang ditandai dengan y_i dan x_{i}, dimana i adalah elemen dalam setiap pasangan observasi. Perbedaan pasangan yang sesuai diperlakukan sebagai variabel, dimana logika dari *paired t-test* dan *one sample t-test* bersifat identik. Persamaan ini kemudian disertakan ke dalam penghitungan *mean of difference* dan *standard deviation of difference* yang ditunjukkan melalui persamaan (2.15) dan (2.16) secara berturut-turut.

$$\bar{d} = \frac{\sum_{i=1}^{n} d_i}{n} \tag{2.15}$$

Rumus 2.15 *Mean of Difference*

$$s_d = \sqrt{\frac{\sum_{i=1}^{n} \left(d_i - \bar{d}\right)^2}{n - 1}} \tag{2.16}$$

Rumus 2.16 *Standard Deviation of Difference*

Berdasarkan penghitungan melalui persamaan (2.15) dan (2.16), terdapat $\bar{d}$ sebagai *mean*, s_d sebagai *standard deviation*, dan n sebagai ukuran sampel. Hasil penghitungan rata-rata dan standar deviasi dari perbedaan digunakan untuk *t statistic*, dimana mencakup pembagian *mean of difference* ($\bar{d}$) dengan *standard error* dari *mean of difference* yang ditandai dengan $s_d/\sqrt{n}$. Berikut terdapat penghitungan *t-statistic* yang ditampilkan pada persamaan (2.17).

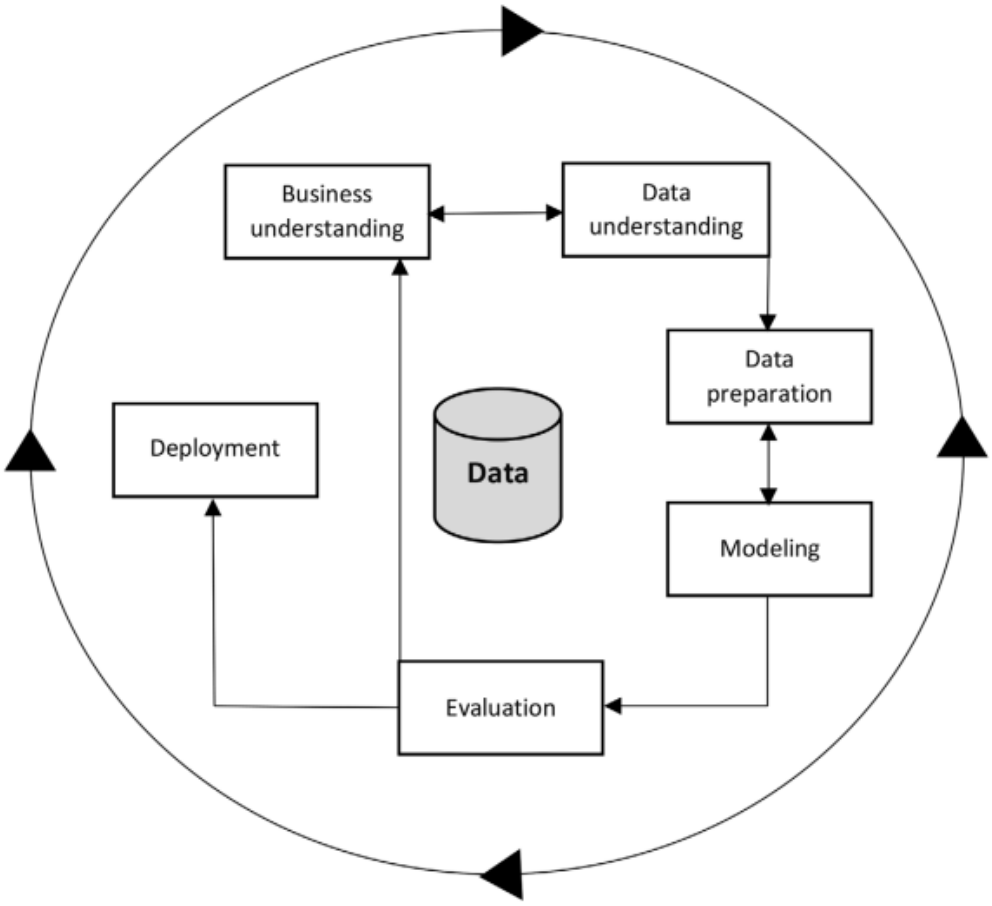
$$t = \frac{\bar{d}}{s_d/\sqrt{n}} \tag{2.17}$$

Rumus 2.17 *t-Statistic*

2.3 *Framework* dan Algoritma

2.3.1 *CRISP-DM Framework*

Cross Industry Standard Process – Data Mining (*CRISP-DM)* merupakan suatu model proses independen yang digunakan untuk *data mining* dalam suatu industri [62]. *CRISP-DM* berdasar pada model siklus hidup *waterfall* dengan proses hierarkis yang mendukung pemberian panduan dalam melaksanakan suatu proyek [63]. *CRISP-DM* telah melakukan pemecahan dan perluasan akan langkah-langkah yang tersedia dalam proposal *Knowledge Discovery in Databases* (*KDD*) yang orisinil menjadi enam langkah, yang dimulai dari *Business Understanding*, *Data Understanding*, *Data Preparation*, *Modeling*, *Evaluation*, hingga *Deployment* [64]. Adapun kerangka kerja yang dimiliki oleh *CRISP-DM* yang menunjukkan langkah-langkah dalam proses *data mining*.



Gambar 2.2 Kerangka Kerja *CRISP-DM*
Sumber: [25]

Adapun penjelasan dari setiap langkah yang dilakukan dalam kerangka kerja *CRISP-DM* yang disajikan dalam Gambar 2.2 [25].

1) *Business Understanding*

Business understanding merupakan langkah yang berfokus dalam memahami tujuan dan ketentuan dalam proyek yang dilihat dari perspektif suatu bisnis perusahaan, dimana pada akhirnya pengetahuan yang diperoleh akan diubah menjadi definisi dari suatu permasalahan pada *data mining* serta rencana proyek awal yang didesain untuk mencapai tujuan yang sudah ditentukan. *Business understanding* merupakan langkah yang memegang peranan penting dalam menentukan tujuan bisnis yang mampu mengarahkan pada proyek selanjutnya.

2) *Data Understanding*

Data understanding merupakan langkah dimana data awal akan dikumpulkan untuk meningkatkan pemahaman suatu perusahaan mengenai data tersebut. Dalam *data understanding*, diperlukan proses identifikasi akan potensi munculnya masalah kualitas data, wawasan terdahulu mengenai data yang bersangkutan, dan kemungkinan hipotesis yang merupakan hasil dari *subset* data yang mampu menampilkan informasi tersirat dari data tersebut.

3) *Data Preparation*

Data preparation merupakan langkah yang melibatkan penugasan akan peran yang spesifik, seperti *data reduction*, data *wrangling* dan *data cleansing*, serta *data transformation* berupa pembuatan variable *dummy* untuk kebutuhan analisis dan pengujian pada langkah selanjutnya.

4) *Modeling*

Modeling merupakan langkah yang mencakup seleksi dan pengembangan dari teknik dan model analitik yang digunakan. Dalam *modeling*, beberapa bagian dari *dataset* seringkali disisihkan untuk kegiatan *training* dan *validation* atas model yang dibuat.

5) *Evaluation*

Evaluation merupakan langkah dimana perusahaan akan melakukan peninjauan dan interpretasi hasil analisis dari kegiatan *modeling* dalam konteks tujuan bisnis dan indikator akan keberhasilan yang sesuai dengan ketentuan yang telah ditetapkan pada langkah awal.

6) *Deployment*

Deployment merupakan langkah akhir dimana hasil pengetahuan yang didapatkan dari data analisis akan diubah ke dalam bentuk serangkaian rekomendasi yang dapat digunakan pada tindakan selanjutnya. Dalam *deployment*, diperlukan pemahaman akan efisiensi dalam menyampaikan hasil analisis terhadap unsur bisnis memegang peran utama dalam keberhasilan suatu proyek analitik.

2.3.2 *Random Forest*

Random Forest adalah metode pembelajaran terintegrasi yang diusulkan oleh Breiman pada tahun 2001 berdasarkan *decision tree* [65]. *Random Forest* menggunakan metodologi *supervised learning* dimana informasi dari *data set* berlabel (*set* pelatihan) diambil untuk memperoleh prediksi dan membangun sebuah model [66]. Dalam pembuatan modelnya, setiap *decision tree* memilih kelas untuk observasi dan prediksi berdasarkan kelas dengan suara terbanyak [67].

Random Forest adalah pengklasifikasi yang terdiri atas kumpulan pengklasfikasi pohon yang terstruktur yang ditandai dengan $\{h(\boldsymbol{x}, \Theta_k), k = 1, \ldots\}$, dimana $\{\Theta_k\}$ adalah vektor-vektor acak yang terdistribusi secara identik dan setiap pohon memberikan satu suara untuk kelas yang paling populer pada *input* $\boldsymbol{x}$ [68]. Proses agregasi pada *Random Forest* dilakukan setelah pembuatan vektor-vektor acak. Dalam proses ini, diberikan sebuah *ensemble* pengklasifikasi $h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_k(\boldsymbol{x})$ dengan *set* pelatihan yang diambil secara acak dari distribusi vektor acak Y**,** **X**, terdapat penghitungan *margin function* yang dilakukan melalui persamaan (2.18) [68].

$$mg(\mathbf{X}, Y) = av_k I(h_k(\boldsymbol{X}) = Y) - max_{j \neq Y}\, av_k I(h_k(\boldsymbol{X}) = j) \quad (2.18)$$

Rumus 2.18 *Margin Function*

Persamaan (2.18) menunjukkan *margin function*, dimana $I(\cdot)$ adalah *indicator function* dan av_k adalah rata-rata, $h_k(\boldsymbol{X}) = Y$ adalah hasil dari klasifikasi, dan $h_k(\boldsymbol{X}) = j$ adalah hasil klasifikasi dengan *j* [69]. *Margin function* mengukur sejauh mana rata-rata suara di **X**, Y untuk kelas yang tepat melebihi rata-rata suara untuk kelas lainnya. Semakin besar *margin* yang diperoleh, maka semakin besar kepercayaan terhadap klasifikasi. Proses penghitungan *generalization* dilakukan setelah memperoleh *margin function*, dimana $P_{\boldsymbol{X},Y}$ menandakan bahwa probabilitas berada di atas ruang **X**, Y, dan $mg(\mathbf{X}, Y)$ adalah *margin function*. Berikut penghitungan *generalization error* yang dilakukan melalui persamaan (2.19) [68].

$$PE^* = P_{\boldsymbol{X},Y}(mg(\mathbf{X}, Y) < 0) \quad (2.19)$$

Rumus 2.19 *Generalization Error*

2.3.3 *K-Nearest Neighbors*

K-Nearest Neighbors (*KNN*) merupakan salah satu *supervised machine learning* yang bersifat *non-parametric* digunakan untuk klasifikasi dan regresi [70]. *K-Nearest Neighbors* diperkenalkan oleh Evelyn Fix dan Joseph Hodges pada tahun 1951 [71]. Algoritma *K-Nearest Neighbors* (*KNN*) bekerja dengan mengukur jarak antara data *input* dengan *k* data terdekat dalam *set* pelatihan [72]. Dalam klasifikasi, *KNN* mencari suara mayoritas dari sejumlah tetangga (*k*) dari sebuah data *input* untuk memilih kelas yang sesuai. Sementara dalam regresi, variabel respon diprediksi dengan merata-ratakan pengamatan dalam tetangga terdekat berdasarkan ukuran kemiripan [70]. Penelitian ini berfokus pada penggunaan *K-Nearest Neighbors Regressor* untuk memprediksi jumlah interaksi pada *channel recommendation* yang dibuat dengan *Cosine distance*. *Cosine distance* merupakan pengukuran yang sering digunakan dalam algoritma pembelajaran mesin untuk menghitung jarak atau kemiripan antara dua titik data [73]. Pada awalnya, dilakukan penghitungan *cosine similarity*

untuk mengukur korelasi antara dua vektor, yang dilakukan melalui persamaan (2.20) [74].

$$\cos(x,\ y) = \frac{x \cdot y}{\|x\| \cdot \|y\|} \tag{2.20}$$

Rumus 2.20 *Cosine Similarity*

Persamaan (2.20) menunjukkan persamaan *cosine similarity*, dimana $(x,\ y)$ menandakan sudut antara dua vektor, $x \cdot y$ menunjukkan *dot product* dari vektor *x* dan *y*, $\|x\| \cdot \|y\|$ menunjukkan panjang dari vektor *x* dan *y*. Proses penghitungan *cosine distance* dilakukan untuk menghasilkan nilai yang bervariasi dari 0 hingga 1. Nilai 0 menandakan bahwa *distance* yang diperoleh dekat sehingga terdapat kemiripan, sedangkan nilai 1 menandakan bahwa *distance* yang diperoleh jauh sehingga tidak terdapat kemiripan. Berikut merupakan penghitungan *cosine distance* yang ditunjukkan dalam persamaan (2.21) [20].

$$cosine\ distance\ =\ 1\ -\ cosine\ similarity \tag{2.21}$$

2.4 *Tools*

2.4.1 PostgreSQL

PostgreSQL merupakan sistem *database object-relational* yang bersifat *open source* yang kuat yang menggunakan dan memperluas bahasa SQL yang dikombinasikan dengan banyak fitur yang menyimpan dan menskalakan *workload* yang paling rumit dengan aman [75]. PostgreSQL memiliki nama yang berasal dari Ingres, yaitu sebuah *database* relasional yang dikembangkan oleh Profesor Michael Stonebraker [76]. PostgreSQL memiliki bahasa pemrograman prosedural bawaan yang bernama bahasa PL/pgSQL. Tujuan desain dari PL/pgSQL adalah untuk membuat bahasa prosedural yang dapat dimuat untuk fungsi, prosedur, dan *trigger*, menambahkan *control structure* ke bahasa SQL, melakukan komputasi yang kompleks, mewarisi semua tipe, fungsi, prosedur, dan operator yang ditentukan pengguna, dapat didefinisikan untuk dipercaya oleh *server*, serta mudah digunakan [77]. PostgreSQL telah mendapatkan reputasi yang kuat karena arsitekturnya yang telah terbukti, keandalan, integritas data, set fitur yang tangguh, ekstensibilitas, dan dedikasi

komunitas *open source* di balik *software* yang memberikan solusi yang berkinerja dan inovatif secara konsisten [75].

2.4.2 DBeaver

DBeaver adalah *tool database* universal yang gratis dan bersifat *open source* untuk para pengembang dan administrator *database* [78]. DBeaver dapat digunakan untuk menghasilkan dan melengkapkan *database* di seluruh pengaturan administrasi *database* yang beragam. DBeaver dapat bekerja dengan *DBMS* secara umum, seperti MySQL, PostgreSQL, MariaDB, SQLite, Oracle, DB2, SQL Server, Sybase, Microsoft Access, Teradata, Firebird, Derby, dan lain sebagainya [79]. Dengan kegunaan sebagai tujuan utamanya, DBeaver menawarkan *User Interface* yang dirancang dan diimplementasikan dengan cermat, berbagai dukungan dalam sumber data *cloud*, standar keamanan perusahaan, dan *multiplatform*, berbagai jenis fitur, serta kemampuan untuk bekerja dengan berbagai *extension* untuk berintegrasi dengan Excel, Git, dan lain sebagainya [80].

2.4.3 Python

Python merupakan bahasa pemorgraman yang kuat dan mudah dipelajari. Bahasa pemograman ini memiliki struktur data yang bersifat *high-level* dan pendekatan yang sederhana namun efektif untuk pemrograman berorientasi objek [81]. Python diciptakan pada akhir tahun 1980an dan pertama kali dirilis pada tahun 1991 oleh Guido van Rossum sebagai penerus bahasa pemrograman ABC [82]. Python dikenal dengan keberagaman *library* yang membantu analisis data dan komputasi [83]. Kemampuan pemrograman dan metodologi *object-oriented* dirancang untuk membantu para pemrogram untuk menulis kode secara ringkas dan logis untuk aplikasi kecil dan besar [84].

2.4.4 Jupyter *Notebook*

Jupyter merupakan salah satu aplikasi web *open-source* gratis yang terkenal pada saat ini di kalangan pengguna dalam menulis dokumen dalam bentuk teks penjelasan, penghitungan, dan visualisasi, serta *live code* beserta hasil eksekusinya [85]. *Notebook* memiliki perbedaan dengan karya ilmiah pada

umumnya dimana *notebook* memungkinkan pengguna untuk berinteraksi secara langsung dengan data dan kode yang dimiliki, serta melakukan pembaruan pada tabel dan diagram dalam tempat yang sama. Selain itu, pengguna dapat menjalankan ulang kode *notebook* setelah melakukan perubahan pada data ataupun algoritma yang berdampak pada hasil akhir *notebook* tersebut [86].

2.4.5 R *Programming Language*

R adalah sebuah bahasa dan *environment* untuk analisis statistik dan visualisasi data. R merupakan proyek *GNU* mirip dengan bahasa dan *environment* S yang awalnya dikembangkan oleh John Chambers dan timnya di Bell *Laboratories* (sebelumnya AT&T, sekarang bagian dari Lucent *Technologies*) [87]. R menyediakan berbagai macam teknik statistik (pemodelan linier dan nonlinier, uji statistik klasik, analisis *time-series*, klasifikasi, *clustering*, dan lain sebagainya), dan teknik grafis, serta sangat mudah dikembangkan [87].

2.4.6 RStudio

RStudio merupakan sebuah *Integrated Development Environment* (*IDE*) untuk R dan Python [88]. RStudio menyediakan sebuah *window* untuk pembuatan *scripts*, mendukung *command entry*, dan termasuk alat visualisasi. Antarmuka RStudio dibagi menjadi 4 bagian, memberikan gambaran umum simultan dari data, perintah, hasil, dan grafik yang dihasilkan. Filosofi RStudio yang dikembangkan dan disediakan oleh RStudio, Inc., yaitu memberdayakan pengguna sehingga dapat menggunakan R secara produktif [89].