

BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian ini menggunakan beberapa penelitian terdahulu yang berkaitan pada penelitian saat ini, yang dapat dijadikan sebagai bahan referensi untuk menganalisis NPL pada industri perbankan menggunakan algoritma *machine learning*. Tabel 2.1 merupakan beberapa penelitian terdahulu yang digunakan.

Tabel 2. 1 Penelitian Terdahulu

No	Judul	Metode	Kesimpulan
1	<i>Predictive Performances of Ensemble Machine Learning Algorithms in Landslide Susceptibility Mapping Using Random Forest, Extreme Gradient Boosting (XGBoost) and Natural Gradient Boosting (NGBoost).</i> Taskin Kavzoglu, Alihan Teke	<i>Random Forest, Extreme Gradient Boosting, dan Natural Gradient.</i>	Berdasarkan penelitian tersebut, algoritma NGBoost menghasilkan kinerja terbaik dengan akurasi keseluruhan sebesar 87.20%. Adapun, XGBoost menghasilkan 82.67% [8].
2	<i>Predicting the Rock Sonic Logs While Drilling by Random Forest and Decision Tree-Based Algorithms.</i> Hany Gamal, Ahmed Saihati, Salaheldin Elkatatny	<i>Random Forest dan Decision Tree.</i>	Algoritma <i>Random Forest</i> lebih unggul dengan nilai akurasi dan korelasi lebih tinggi dibanding <i>Decision Tree</i> . [9].
3	<i>Research on the application of Decision Tree and Random Forest Algorithm in the main transformer fault evaluation.</i> Chenmeng Zhang, Can Hu, Shijun Xie, Shuping Cao	<i>Random Forest dan Decision Tree.</i>	Algoritma <i>Random Forest</i> lebih unggul dengan akurasi sebesar 88.2% dibanding <i>Decision Tree</i> dengan akurasi 86.3% [10].
4	<i>Comparison of Prediction Models for Acute Kidney Injury Among Patients with Hepatobiliary Malignancies Based on XGBoost and LASSO-Logistic Algorithms.</i> Yunlu Zhang, Yimei Wang, Jiarui Xu, Bowen Zhu,	<i>XGBoost dan model Lasso-logistik.</i>	Model <i>XGBoost</i> yang dikembangkan memiliki nilai akurasi yang lebih tinggi pada kanker hati sebesar 82% dan 85% pada kanker empedu [11].

No	Judul	Metode	Kesimpulan
	Xiaohong Chen Xiaoqiang Ding, Yang Li		
5	<i>Comparison of the Performance Results of C4.5 and Random Forest Algorithm in Data Mining to Predict Childbirth Process.</i> Muhasshanah, Mohammad Tohir, Dewi Andariya Ningsih, Neny Yuli Susanti, Astik Umiyah, Lia Fitria	<i>Random Forest</i> dan C4.5.	Berdasarkan penelitian yang telah dilaksanakan, algoritma C4.5 menghasilkan akurasi yang lebih tinggi sebesar 96% dibanding algoritma <i>Random Forest</i> , yaitu 95% [12].
6	<i>A Comparative Study of Machine Learning Approaches for Non-Performing Loan Prediction with Explainability</i> Sefik Ilkin Serengil, Salih Imece, Ugur Gurkan Tosun, Ege Berk Buyukbas, Bilge Koroglu	<i>Logistic Regression, Random Forest Bagging Classifier, Support Vector Machines, XGBoost LightGBM, dan LSTM.</i>	Berdasarkan penelitian yang telah dilaksanakan, algoritma <i>XGBoost</i> memberikan hasil yang terbaik dalam memprediksi NPL dengan akurasi 90% [13].
7	<i>Comparative Performance Analysis of Machine Learning Algorithms for Non Performing Loan Prediction</i> Tanmay Chaturvedi, Sukanta Halder, Nilanjan Das, Uppalapati Sudheer kumar, SK Bittu	<i>Random Forest, Gradient Boost, XGBoost, LSTM, Decision Tree, Gaussian Naive Bayes, LightGBM, AdaBoost, Vector Machines, dan Logistic Regression.</i>	Berdasarkan penelitian yang telah dilaksanakan, algoritma <i>Random Forest</i> menghasilkan akurasi yang tertinggi sebesar 89.11% [14].
8	<i>Data Augmentation for Occlusion-Robust Traffic Sign Recognition Using Deep Learning</i> Andrew Dineley, Friska Natalia, Sud Sudirman	<i>AlexNet, VGG19, ResNet50, dan GoogleNet.</i>	Penelitian ini menghasilkan model <i>GoogleNet</i> yang menunjukkan peningkatan hasil akurasi tertinggi sebesar 17%. Adapun, pada penelitian ini menggunakan <i>data splitting</i> 80:20 yang dijadikan landasan pada penelitian ini [15].

Tabel 2.1 menjelaskan mengenai delapan penelitian terdahulu yang dapat dijadikan referensi sebagai pendukung pada penelitian ini. Berbagai penelitian terdahulu pada Tabel 2.1, memiliki beberapa masukan dimana diperlukan pengumpulan data yang lebih lengkap dan menggunakan data yang beragam agar dapat mengatasi masalah ketidakseimbangan data pada penelitian, sehingga dapat memperoleh akurasi pada model yang akan dihasilkan nantinya.

Algoritma yang digunakan pada penelitian-penelitian terdahulu dalam melakukan klasifikasi kelas target pada setiap penelitian, yaitu *Decision Tree*, *Random Forest*, *C4.5*, *XGBoost*, *LASSO-Logistic*, *Logistic Regression*, *SVM*, *Bagging Classifier*, *LSTM*, *LightGBM*, *Gaussian Naïve Bayes*, *Natural Gradient Boosting* (NGBoost), *VGG19*, *ResNet50*, dan *AlexNet*. Adapun, dari berbagai algoritma tersebut algoritma *NGBoost*, *Random Forest*, *XGBoost*, *C4.5*, *LightGBM*, dan *GoogleNet* memiliki akurasi yang paling tinggi. Oleh karena itu, algoritma *XGBoost*, *Random Forest*, dan *Decision Tree* akan digunakan pada penelitian ini.

Penelitian ini bertujuan untuk membandingkan performa kinerja algoritma *machin learning Random Forest*, *Decision Tree*, dan *XGBoost* yang dipilih dari penelitian terdahulu untuk menganalisis *non-performing loan* (NPL) pada industri perbankan dengan menggunakan keseluruhan kategori kredit.

2.2 Teori Penelitian

2.2.1 *Non-Performing Loan (NPL)*

Non-Performing Loan (NPL) adalah suatu istilah dalam dunia perbankan yang dikategorikan ketika kredit bermasalah. Dalam hal ini kredit bermasalah merupakan pinjaman yang tidak dapat dilunasi oleh nasabah sesuai dengan jadwal pembayaran atau ketentuan yang telah disepakati bersama pihak bank atau nasabah tersebut dinyatakan gagal dalam memenuhi kewajiban pembayarannya [16]. Pada umumnya, kredit bermasalah dikategorikan jika nasabah telah menunggak lebih dari 180 hari atau sekitar 6 bulan menunggak dari jumlah hari kesepakatan tersebut [17]. Adapun, berdasarkan peraturan dari Bank Indonesia suatu kredit terdiri dari beberapa kategori, yaitu lancar atau kategori 0, dalam perhatian khusus (DPK) atau kategori 1, kurang lancar atau kategori 2, diragukan atau kategori 3, dan bermasalah atau dinyatakan telah macet yaitu kategori 4. Pada kategori dalam perhatian khusus (DPK), berarti nasabah terlambat membayar kesepakatan pinjaman selama 30-90 hari. Kategori kurang lancar berarti nasabah mengalami keterlambatan pembayaran selama 90-120 hari. Kategori diragukan berarti nasabah mengalami keterlambatan pembayaran selama 120-180 hari. Kategori macet (NPL) berarti

nasabah telah mengalami keterlambatan pembayaran yang telah melebihi 180 hari atau sekitar 6 bulan menunggak [17].

Tingginya tingkat NPL dapat berisiko terhadap portofolio suatu perbankan. Adapun, nasabah perlu untuk menjaga reputasi kredit dengan menepati perjanjian terkait kewajiban pembayaran agar terhindar dari status kredit macet atau NPL. Tingginya status NPL pada sektor perbankan dapat memberikan berbagai dampak negatif, seperti tingginya likuiditas bank dimana bank akan memiliki potensi kesulitan dalam memberikan kredit kepada nasabah, sulit memperoleh laba atau tidak memperoleh keuntungan dari kredit yang diberikan kepada nasabah, dan menyebabkan modal dari suatu perbankan berkurang. Disamping itu, penyebab terjadinya kredit bermasalah dari nasabah dikarenakan adanya perubahan kebijakan dari pemerintah atau turunnya penjualan dari usaha yang dijalankan oleh para nasabah [18]. Jika kasus NPL semakin hari semakin tinggi maka dapat menimbulkan krisis keuangan, seperti yang terjadi pada krisis moneter Asia di tahun 1997-1998 dimana beberapa bank di Indonesia mengalami kasus NPL yang mencapai lebih dari 30% dan berakibat pada bangkrutnya bank tersebut. Oleh karena itu, pengelolaan dan analisis terhadap NPL merupakan hal yang krusial untuk menjaga stabilitas perbankan [19].

2.2.2 Data Mining

Data mining merupakan suatu proses eksplorasi dan analisis data dengan menggabungkan teknik statistik serta *machine learning* yang berjumlah besar dengan tujuan untuk menemukan pola, hubungan, dan informasi berharga untuk membuat keputusan yang lebih baik. Terdapat beberapa proses atau tahapan pada *data mining*, yaitu tahapan pengumpulan data dari berbagai sumber atau *database*, tahapan *data preparation* atau *data cleaning* dilakukan penghapusan data yang tidak relevan atau hilang dan terduplikat, tahapan *data integration* dengan menggabungkan beberapa data untuk memperoleh *dataset* yang baru, tahapan *selection* dengan memilih data yang relevan agar sesuai dengan fokus penelitian, tahapan *data transformation* dimana mengubah data menjadi bentuk yang sesuai seperti normalisasi atau agregasi, tahapan *data mining* dimana menerapkan model *machine learning* untuk memperoleh suatu

informasi atau pola yang tersembunyi, dan tahapan *knowledge presentation* dimana hasil proses *data mining* akan disajikan melalui visualisasi. Terdapat beberapa metode pada *data mining*, yaitu metode klasifikasi dimana metode ini mengkategorikan beberapa data ke dalam suatu kelompok tertentu, metode regresi dimana metode yang memprediksi suatu nilai yang bersifat kontinu, metode *clustering* dimana metode ini akan mengelompokkan data tanpa label atau kategori yang sudah ada, dan metode asosiasi yaitu metode yang mencari pola atau hubungan dari suatu item dalam data [20].

2.2.3 *Machine Learning*

Machine learning (ML) adalah suatu cabang dari *artificial intelligence* yang memungkinkan komputer untuk mempelajari data dan berfokus pada pengembangan model dan algoritma untuk membuat suatu prediksi dan keputusan. Dengan menerapkan *machine learning* maka model atau algoritma dapat mengenali pola yang ada pada data dan berdasarkan pola yang ditemukan akan digunakan untuk memprediksi dan membuat keputusan. Terdapat beberapa jenis pada *machine learning*, yaitu *supervised learning* dimana sistem akan menggunakan data yang telah berlabel untuk memprediksi hasil untuk data yang belum berlabel. Terdapat dua kategori pada *supervised learning*, yaitu klasifikasi dan regresi dimana kedua kategori ini terdiri dari beberapa algoritma, yaitu *Linear Regression*, *Logistic Regression*, *Decision Tree*, *Support Vector Machines (SVM)*, *K-Nearest Neighbors (KNN)*, *Naïve Bayes*, dan *Random Forest*. Adapun, *unsupervised learning* merupakan sistem yang bekerja dengan data yang tidak berlabel untuk memperoleh pola atau struktur pada data. *Unsupervised learning* terdiri dari beberapa kategori, yaitu *clustering*, *association rule mining*, dan *dimensionality reduction*. Kemudian terdapat *reinforcement learning* dimana sistem belajar melalui *error* atau *trial* dengan menerima *feedback* dari penghargaan atau hukuman dari suatu tindakan. Selain itu, terdapat *semi-supervised learning* yang menggunakan beberapa data yang berlabel dan tidak berlabel untuk melatih model dan *self-supervised learning* yang mempelajari data dengan menggunakan bagian dari data sendiri untuk memperoleh label dan biasanya mempelajari data tanpa adanya pengawasan yang eksplisit [21].

2.2.4 *Supervised Learning*

Supervised learning adalah jenis algoritma *machine learning* yang melatih model menggunakan *dataset* yang terlabeli, dimana setiap data yang akan digunakan sudah berlabel, sehingga model akan mempelajari hubungan dari *input* dan *output* data agar dapat memperoleh prediksi yang akurat untuk data baru yang tidak terlabeli. Terdapat beberapa tahapan pada *supervised learning*, yaitu *data training* dimana model akan dilatih dengan data yang terlabeli yang terdiri dari bagian fitur dan *target*. Tahapan yang kedua adalah *model training* dimana algoritma yang digunakan akan menemukan pola dan hubungan antar fitur dan *target* pada *data training* agar dapat meminimalisir kesalahan prediksi nantinya. Tahapan ketiga adalah evaluasi model menggunakan *data testing* untuk mengukur akurasi model dalam melakukan prediksi data baru. Tahapan terakhir adalah prediksi dimana model yang telah dievaluasi akan memprediksi label dari data baru. Terdapat beberapa algoritma pada *supervised learning*, yaitu *Linear Regression*, *Logistic Regression*, *Decision Tree*, *Support Vector Machines (SVM)*, *K-Nearest Neighbors (KNN)*, *Naïve Bayes*, dan *Random Forest* [22].

2.2.5 *Ensemble Learning*

Ensemble Learning adalah salah satu cabang dari *machine learning* dengan menggabungkan beberapa model untuk meningkatkan akurasi atau performa prediksi dan menghasilkan keputusan yang lebih akurat. Dalam hal ini, *ensemble learning* akan memperbaiki kesalahan satu model dengan model lainnya dan dapat mengurangi *overfitting*, *underfitting*, ataupun *variance* yang terjadi pada model tunggal. Terdapat beberapa jenis *ensemble learning*, yaitu *Bootstrap Aggregating (Bagging)* yang merupakan model dengan menggunakan teknik *sampling* untuk subset data berbeda agar dapat membuat banyak model secara paralel, contohnya algoritma *Random Forest*. Kemudian, terdapat *Boosting* dengan algoritma seperti *Gradient Boosting*, *XGBoost*, dan *AdaBoost*. *Boosting* merupakan jenis *ensemble learning* yang melibatkan pembuatan model yang berurutan yang berfokus pada *misclassifications* dari model sebelumnya. Selain itu, terdapat *Stacking* yang menggabungkan hasil prediksi beberapa model seperti *Boosting* dan *Bagging* untuk mengoptimalkan

keputusan akhir. Kemudian, terdapat *Voting* dimana model dilatih dan akan dilakukan *voting* pada bagian keputusan akhirnya berdasarkan rata-rata probabilitas atau mayoritas suara [23].

2.2.6 *Encoding*

Encoding adalah teknik untuk mengubah data yang bersifat kategorikal ke data numerik untuk digunakan oleh algoritma *machine learning*. Terdapat beberapa algoritma yang tidak dapat memproses data kategorikal, seperti regresi dan *Decision Tree*, sehingga memerlukan *encoding* untuk mengkonversi data ke bentuk numerik. Terdapat dua jenis *encoding* yang biasanya digunakan pada beberapa penelitian, yaitu *One-Hot Encoding* yang bekerja dengan mengubah data kategorik ke dalam variabel dan mengisi data tersebut dengan nilai 0 dan 1. Kemudian, *Label Encoding* bekerja dengan cara memberikan label numerik pada kategori, misalnya kategori warna biru dan hijau diberi label 0 dan 1 [24].

2.2.7 *Log-Transform*

Log-Transform adalah teknik untuk mengubah distribusi data yang tidak normal atau miring (*skewed*) menjadi distribusi data yang normal. Teknik *Log-Transform* biasanya digunakan untuk menangani *outlier* mengurangi *skewness* pada data, sehingga distribusi data dapat menjadi normal dan membuat model menjadi lebih stabil serta memiliki nilai akurasi yang tinggi [25].

2.2.8 *Cross-Validation*

Cross-Validation adalah salah satu teknik untuk menilai kinerja atau performa model *machine learning* yang digunakan pada penelitian dengan melakukan pembagian ke dalam beberapa *fold* atau subset data. Pada teknik ini, model akan dilatih dan diuji ke dalam beberapa *fold* agar dapat memastikan model diuji pada data yang berbeda dari data pelatihan. Dengan melakukan *cross-validation* maka model yang akan dibangun dapat menghindari terjadinya *overfitting* dan menghasilkan model yang lebih stabil. Terdapat beberapa jenis *cross-validation*, yaitu *K-Fold Cross Validation*, *Stratified K-Fold*, dan *Leave-One-Out Cross Validation* (LOO-CV) [26].

2.2.9 *GridSearchCV*

GridSearchCV adalah salah satu teknik yang digunakan untuk melakukan *hyperparameter tuning* pada model *machine learning* dengan cara mencari kombinasi yang paling bagus dari parameter model agar memberikan performa model yang terbaik. CV yang ada pada *GridSearchCV* didasarkan dari *cross-validation* dimana dilakukan pembagian data agar model yang diuji dapat meminimalisir terjadinya *overfitting* dan dapat memperoleh model yang stabil atau terbaik. Penerapan teknik ini diawali dengan memilih algoritma *machine learning* yang akan digunakan dan kemudian harus menentukan nilai untuk berbagai parameter yang akan dicoba. Setelah itu, *GridSearchCV* akan melakukan *cross-validation* pada setiap kombinasi parameter agar dapat memperoleh parameter mana yang akan menghasilkan akurasi terbaik. Langkah terakhir adalah *GridSearchCV* akan memilih model terbaik dari perhitungan *cross-validation* [27].

2.2.10 *Confusion Matrix*

Confusion Matrix adalah matriks yang digunakan untuk melakukan evaluasi terhadap performa model klasifikasi dan memberikan suatu gambaran mengenai berapa banyak prediksi yang benar dan salah dari setiap kelas yang ada pada *dataset*. Melalui *confusion matrix*, beberapa metrik lain seperti *accuracy*, *precision*, *recall*, dan *F1-Score* dapat diketahui nilainya. Dengan menggunakan *confusion matrix* maka dapat menunjukkan performa model yang lebih lengkap dan menunjukkan kesalahan yang dilakukan model, seperti adanya *misclassification* pada kelas positif dan negatif [28].

Confusion Matrix

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

Gambar 2. 1 Tabel *Confusion Matrix*

Sumber: *Glass Box*, Februari 2019 [29]

Gambar 2.1 menunjukkan gambar tabel *confusion matrix*, dimana TP merupakan model melakukan klasifikasi dengan benar sebagai positif, FP merupakan model salah dalam melakukan klasifikasi sebagai positif, TN merupakan model melakukan klasifikasi dengan benar sebagai negatif, dan FN merupakan model yang salah dalam melakukan klasifikasi sebagai negatif. Berdasarkan *confusion matrix* tersebut maka dapat dilakukan perhitungan lain seperti [29]:

- a) *Accuracy*, metrik yang digunakan untuk mengukur berapa banyak prediksi yang benar dari keseluruhan prediksi yang dibuat oleh model. Akurasi dihitung dengan menambahkan *True Positive* dan *True Negative* dibagi dengan penjumlahan dari *True Positive*, *True Negative*, *False Positive*, dan *False Negative*. Berikut rumus dari *accuracy*:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

Rumus 2. 1 Rumus *Accuracy* [28]

- b) *Precision*, metrik yang digunakan untuk mengukur berapa banyak prediksi yang benar-benar positif. *Precision* dihitung dengan membagi *True Positive* dengan total penjumlahan dari *True Positive* dan *False Positive*. Berikut rumus dari *precision*:

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

Rumus 2. 2 Rumus *Precision* [28]

- c) *Recall*, metrik yang digunakan untuk mengukur berapa banyak dari keseluruhan kasus positif yang berhasil dideteksi oleh model. *Recall* dihitung dengan membagi *True Positive* dengan total penjumlahan dari *True Positive* dengan *False Negative*. Berikut rumus dari *recall*:

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

Rumus 2. 3 Rumus *Recall* [28]

- d) *F1-Score*, metrik yang digunakan untuk menggabungkan metrik *recall* dan *precision* agar memperoleh metrik yang lebih seimbang. *F1-Score* dihitung dengan mengalikan presisi dengan *recall* dan

hasilnya akan dibagi dengan jumlah dari presisi ditambah dengan *recall*. Setelah itu, hasil yang diperoleh akan dikali dengan nilai 2. Berikut rumus dari *F1-Score*:

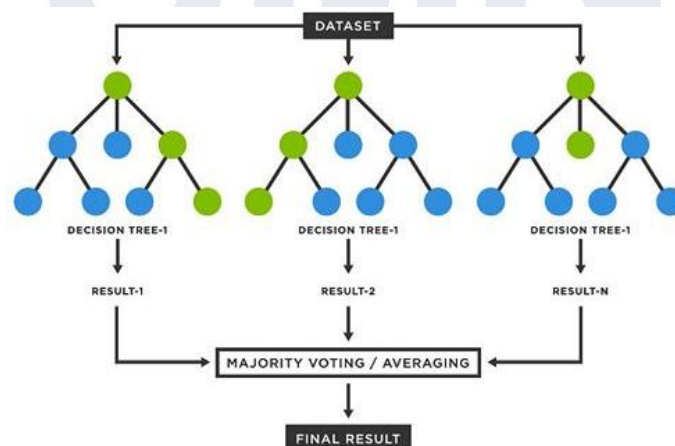
$$F1\text{-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Rumus 2. 4 Rumus *F1-Score* [28]

2.3 Framework dan Algoritma Penelitian

2.3.1 Random Forest

Random Forest merupakan algoritma yang hampir mirip dengan *Decision Tree* dan terdiri dari beberapa *Decision Tree*. Algoritma ini termasuk ke dalam kategori *ensemble learning* yang digunakan untuk melakukan prediksi yang lebih akurat dan stabil. Akan tetapi, algoritma ini juga dapat digunakan pada kategori *supervised learning* untuk mengatasi masalah regresi dan klasifikasi. Algoritma ini memiliki beberapa keuntungan, seperti dapat mengurangi risiko *overfitting*, efektif dalam menangani *missing value*, dan sangat sesuai untuk menangani data yang berukuran besar. Namun, terdapat kekurangan pada algoritma ini dimana model ini cenderung menjadi lebih kompleks diinterpretasikan dan membutuhkan waktu pelatihan yang lama jika memiliki jumlah pohon yang banyak. Pada algoritma ini, model akan memilih secara *random* fitur untuk diamati dan dilakukan Pembangunan pohon keputusan untuk dihitung nilai rata-rata hasilnya [30].



Gambar 2. 2 Cara Kerja *Random Forest*

Sumber: Babylonian Journal of Machine Learning, 2024 [30]

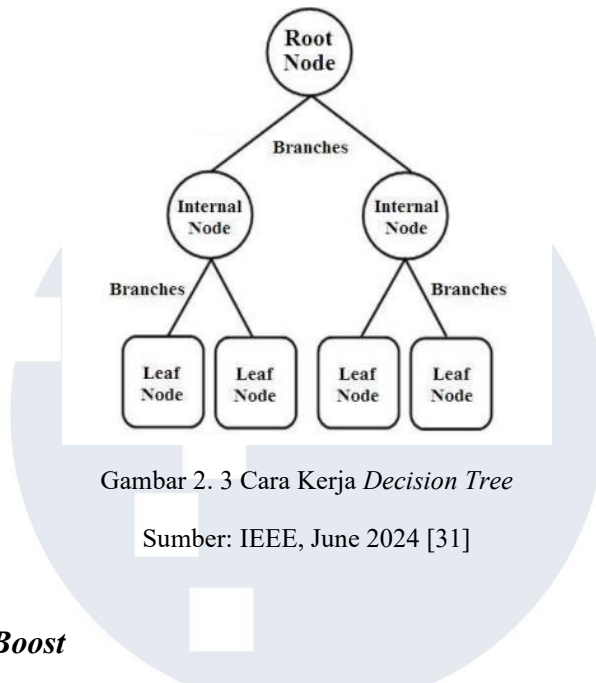
Gambar 2.2 menjelaskan mengenai cara kerja algoritma *Random Forest* yang dimulai dengan memasukkan *dataset* untuk diproses oleh beberapa *Decision Tree* secara paralel. Setelah itu, beberapa *Decision Tree* tersebut akan dilatih dengan subset data yang berbeda menggunakan teknik *bootstrapping*. Setelah itu, setiap pohon yang terbentuk akan membagi data dengan menggunakan subset acak dari fitur dan akan memperoleh hasil prediksi berupa kelas atau nilai tergantung pada masalah penelitian, yaitu regresi atau klasifikasi. Kemudian, setelah berbagai pohon tersebut memperoleh prediksi atau klasifikasi maka hasil yang diperoleh dari setiap pohon akan digabung menggunakan metode *majority voting* (klasifikasi) atau *averaging* (regresi) [30].

2.3.2 *Decision Tree*

Decision Tree merupakan algoritma yang digunakan untuk masalah regresi dan klasifikasi, dimana data yang dikelompokkan akan menjadi keputusan yang didasarkan pada fitur yang ada, sehingga dapat digunakan untuk memprediksi hasil dan membuat keputusan yang lebih baik yang didasarkan pada data. *Decision Tree* yang digunakan bertipe C4.5 yang merupakan algoritma untuk mengatasi masalah klasifikasi dengan tujuan menghasilkan pohon keputusan yang dapat memprediksi kategori suatu objek berdasarkan beberapa atribut yang ada. Algoritma ini memiliki beberapa kelebihan, yaitu mudah diinterpretasikan dan dipahami, tidak membutuhkan skala fitur seperti algoritma SVM dan KNN dimana algoritma ini tidak membutuhkan normalisasi, dan dapat menangani data numerikal serta kategorikal. Namun, algoritma ini memiliki beberapa kekurangan juga, yaitu rentan terhadap *overfitting*, sensitive terhadap perubahan kecil pada data, dan bias terhadap fitur yang memiliki banyak kategori [31].

Gambar 2.3 menunjukkan proses cara kerja dari algoritma *Decision Tree*, dimana data akan dibagi berdasarkan kondisi pada *root node* yang merupakan tempat data dibagi pertama kali yang bertugas untuk memilih fitur. Pada gambar tersebut, jika suatu kondisi pada *root node* adalah *no* maka data akan diproses pada cabang kiri. Setelah itu, pada bagan *decision node* akan

dilakukan pembagian data juga hingga ke bagian *leaf nodes*. *Decision node* bertugas untuk memecah data berdasarkan suatu kondisi. Pada bagian *leaf nodes*, hasil akhir dari klasifikasi atau prediksi akan diperoleh oleh model *Decision Tree* [31].



Gambar 2. 3 Cara Kerja *Decision Tree*

Sumber: IEEE, June 2024 [31]

2.3.3 *XGBoost*

XGBoost merupakan algoritma pada kategori *ensemble learning* yang berdasarkan pada metode *Gradient Boosting* dan dikenal sebagai salah satu algoritma yang efisien pada *dataset* yang kompleks atau berukuran besar. Algoritma ini memiliki beberapa kelebihan, seperti memberikan nilai akurasi yang tinggi, efisien terhadap dapat menangani *missing value* dengan cara yang efektif, dan terdapat beberapa parameter yang dapat disesuaikan untuk mengoptimalkan model. Namun, *XGBoost* juga memiliki kekurangan pada waktu pelatihan yang dapat memakan waktu lebih lambat jika diterapkan pada *dataset* yang berukuran kecil [32].

Cara kerja algoritma *XGBoost* yang pertama adalah membangun *Decision Tree* menggunakan metode *Gradient Boosting* yang berfokus pada kesalahan dari pohon sebelumnya. Setelah itu, model akan menghitung kesalahan pada setiap iterasi sehingga akan dibuat pohon baru yang dapat mengatasi kesalahan tersebut. Kemudian, untuk mencegah *overfitting* maka *XGBoost* akan mengimplementasikan regularisasi menggunakan L1 dan L2 *regularization*. Selain itu, model akan menggunakan *parallel processing* dan

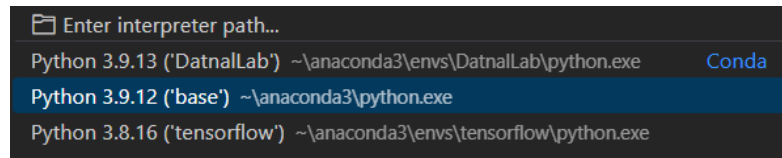
- c) *Data Preparation*, tahapan yang bertujuan untuk membersihkan dan memproses data agar dapat dianalisis lebih lanjut. Tahap ini terdiri dari beberapa proses, seperti *data cleaning* dengan menghapus data yang terduplikat atau menangani *missing value*, dan mengubah format data.
- d) *Modeling*, tahapan yang bertujuan untuk membangun model *machine learning* sesuai dengan tujuan atau permasalahan bisnis. Tahap ini terdiri dari beberapa proses, seperti membangun model menggunakan algoritma klasifikasi, regresi, atau *clustering*.
- e) *Evaluation*, tahapan yang bertujuan untuk menilai performa atau kinerja model yang dibangun pada tahap *modeling* dengan mengevaluasi apakah model tersebut sudah memenuhi tujuan bisnis atau belum. Tahap ini terdiri dari beberapa proses, seperti mengukur akurasi dan performa.
- f) *Deployment*, tahapan yang bertujuan untuk mengimplementasikan model yang telah dievaluasi ke dalam dunia nyata bisnis. Pada tahap ini, model akan memberikan hasil berupa prediksi atau rekomendasi dan akan diimplementasikan melalui pembuatan laporan, *dashboard*, atau sistem.

2.4 Tools dan Software Penelitian

2.4.1 Visual Studio Code

Visual studio code merupakan *platform code editor* yang dikembangkan oleh *Microsoft* yang dapat digunakan untuk proses *development* aplikasi. *Visual studio code* dapat juga digunakan untuk mengedit *source code* dengan bahasa pemrograman, seperti *JavaScript*, *Node.js*, dan lainnya. Selain itu, *platform code editor* ini mendukung berbagai bahasa pemrograman lainnya, seperti *PHP*, *Python*, *Java*, dan *.NET*. *Visual studio code* memiliki beberapa kelebihan, seperti dapat digunakan pada berbagai *platform* seperti *Mac OS*, *Linux*, dan *Windows*. Kemudian, *visual studio code* merupakan *platform* yang tidak berbayar,

memiliki performa yang cepat, dan mendukung berbagai bahasa pemrograman. Adapun, pada *visual studio code* terdapat *Anaconda* yang dapat diintegrasikan dengan *visual studio code* yang dapat mendukung penelitian ini seperti pada Gambar 2.5. Kemudian, *visual studio code* memiliki beberapa fitur, seperti *debugging*, *syntax highlighting*, integrasi *Git*, dan penyelesaian otomatis [34].



Gambar 2. 5 Integrasi Anaconda

2.4.2 *Python*

Python merupakan bahasa pemrograman yang digunakan untuk membangun *website*, *software*, aplikasi, *data science*, melakukan otomatisasi pada tugas, dan melakukan analisis pada data serta *artificial intelligence*. Bahasa pemrograman ini mudah digunakan dan menjadi bahasa pemrograman yang populer dan banyak digunakan. Berdasarkan survei pengembang *Stack Overflow* di tahun 2022, *Python* menjadi bahasa pemrograman yang populer keempat [35].

2.4.3 *Microsoft Excel*

Microsoft excel merupakan *software* yang berbentuk *spreadsheet* yang dikembangkan oleh Microsoft. *Software* ini digunakan untuk melakukan analisis seperti manipulasi data numerik, mengorganisir, dan memvisualisasikan data ke dalam bentuk grafik, tabel, dan rumus matematis dan statistika. Terdapat beberapa rumus umum penting pada Microsoft Excel, yaitu *Sum*, *If*, *Average*, *Max* dan *Min*, serta *Count* [36].

2.4.4 *Streamlit*

Streamlit merupakan merupakan *framework open-source Python* untuk pada *data scientist* atau para *AI/ML engineers* untuk mengembangkan aplikasi web interaktif. *Framework* ini dapat digunakan untuk analisis data, visualisasi data, dan *machine learning* secara cepat dan mudah. Adapun,

terdapat beberapa fitur lainnya dari *streamlit*, yaitu mudah diintegrasikan dengan beberapa *library Python* seperti Pandas, Matplotlib, NumPy, dan lainnya, serta dapat dengan cepat untuk membangun *prototype* aplikasi dalam waktu yang singkat [37].



UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA