Jurnal Politeknik Caltex Riau
Terbit Online pada laman https://jurnal.pcr.ac.id/index.php/jkt/
| e- ISSN : 2460-5255 (Online) | p- ISSN : 2443-4159 (Print) |

Model Prediksi Kemenangan Tim dalam Game *League of Legend* Menggunakan Algoritma Decision Tree

Green Arther Sandag[1]

[1]Universitas Klabat,Manado, Fakultas Ilmu Komputer, email:greensandag@unklab.ac.id

[1] Abstrak

Game online berkembang sangat pesat karena didukung dengan perkembangan ponsel cerdas yang semakin banyak digunakan. Salah satu game online yang yang banyak dimainkan adalah League of Legend. Kemenangan suatu tim dapat ditentukan oleh penggunaan hero maupun strategi pemain, dengan penggunaan machine learning dapat membantu pemain agar mendapat kemenangan dalam tim, maka pada penelitian ini peneliti membuat model dalam memprediksi kemenangan tim pada game league of legend dengan menggunakan algoritma decision tree. Dalam penelitan ini dataset yang digunakan adalah 50.000 data, dibagi menjadi 80% training dan 20% testing. Hasil dari penelitan ini menunjukan algoritma decision tree memiliki performa yang paling baik di antara algoritma lainnya untuk memprediksi kemenangan dengan hasil 96.42% accuracy, 97.74% recall Team 1, 95.06% recall Team 2, 95.31% precision Team 1, 97.62% precision Team 2 dan 0.157 RMSE untuk sedangkan untuk hasil 10-fold cross validation memiliki hasil 96.24% accuracy, 97.34 % recall Team 1, 95.11% recall Team 2, 95.33% precision Team 1, 97.21% precision Team 2, dan 0.161 RMSE.

Kata kunci: *prediksi, game, algoritma decision tree, RMSE*

[2] Abstract

Online games are growing very rapidly because they are supported by the development of smart phones which are increasingly being used. One of the most widely played online games is League of Legend. The victory of a team can be determined using heroes and player strategies, machine learning can help players to win in the team, so in this study the researchers created a model in predicting team victory in the league of legend game using the decision tree algorithm. In this research, the dataset that we use around 50,000 data, divided into 80% training and 20% testing. The results of this research show that the decision tree algorithm has the best performance among other algorithms to predict victory with the results of 96.42% accuracy, 97.74% recall Team 1, 95.06% recall Team 2, 95.31% precision Team 1, 97.62% precision Team 2 and 0.157 RMSE while for 10-fold cross validation results have 96.24% accuracy, 97.34% recall Team 1, 95.11% recall Team 2, 95.33% precision Team 1, 97.21% precision Team 2, and 0.161 RMSE.

Keywords: *prediction, game, decision tree algorithm, RMSE*

Dokumen diterima pada 17 Februari, 2021
Dipublikasikan pada 31 Mei, 2021

1. Pendahuluan

Kemajuan teknologi adalah sesuatu yang tidak bisa dihindari dalam kehidupan ini, karena kemajuan teknologi akan berjalan sesuai dengan kemajuan ilmu pengetahuan [1]. Teknologi adalah alat atau barang-barang dari keseluruhan sarana yang digunakan untuk kelangsungan dan kenyamanan hidup manusia. Teknologi sudah berkembang sangat cepat dari tahun ke tahun. Perkembangan teknologi yang paling tampak adalah telepon, komputer dan internet. Ketiga teknologi ini telah membantu memperkecil hambatan pada komunikasi secara bebas dan global.

Pada era modernisasi sekarang perkembangan teknologi sudah semakin canggih, perkembangan teknologi juga telah diiringi oleh kemajuan teknologi internet. Sekarang ini sudah ada teknologi smartphone yang didalamnya terdapat aplikasi game online yang mudah sekali untuk didownload dan dimainkan [2]. *Game online* berkembang sangat pesat karena didukung dengan perkembangan ponsel cerdas yang semakin banyak digunakan. *Game online* banyak diminati oleh berbagai kalangan mulai dari anak-anak, remaja sampai orang dewasa pun senang bermain game online.

Multiplayer Online Battle Arena (MOBA) adalah salah satu genre pada video game, game bergenre MOBA dimainkan secara berkelompok membentuk dua tim yang berbeda yang bertujuan untuk saling menghancurkan markas tim lawan [3]. *League of Legends (LoL*) merupakan salah saatu game bergenre MOBA dengan jumlah pemain terbanyak didunia [4].

Masalahnya kebanyakkan pemain mengeluh karena *ranked match* mereka terus menurun akibat dari kekalahan dalam permainan. Berdasarkan kasus tersebut peneliti mendapatkan beberapa factor penyebab kekalahan dari permainan mereka. Contoh permasalahan tersebut antara lain: Tidak dapat memaksimalkan perfoma dari *hero* yang dimiliki, strategi kurang tepat, anggota tim banyak yang *noob*.

Decision Tree atau pohon keputusan adalah alat pendukung keputusan. Penggunaan *Decision Tree* ini umunya dalam riset, khususnya dalam analisis keputusan. Tujuan dalam menggunakan *Decision Tree* untuk membantu mengidentifikasi strategi yang paling mungkin untuk mencapai tujuan dan merupakan alat yang populer dalam *machine learning* [5].

Algoritma merupakan salah satu yang digunakan untuk melakukan klasifikasi, segmentasi atau pengelompokan dan bersifat prediktif. Dasar algoritma *decision tree* adalah pembentukan pohon keputusan. Cabang-cabang pohon keputusan merupakan pertanyaan klasifikasi dan daun-daunnya merupakan kelas-kelas atau segmen-segmennya [6]. Prediksi kemenangan dari game ini adalah suatu pandangan yang jarang dilakukan oleh player tapi paling tidak, pada saat permainan dimulai, player dapat melakukan kalkulasi dalam meraih kemenangan dengan mememprediksi kemenangan dalam game ini.

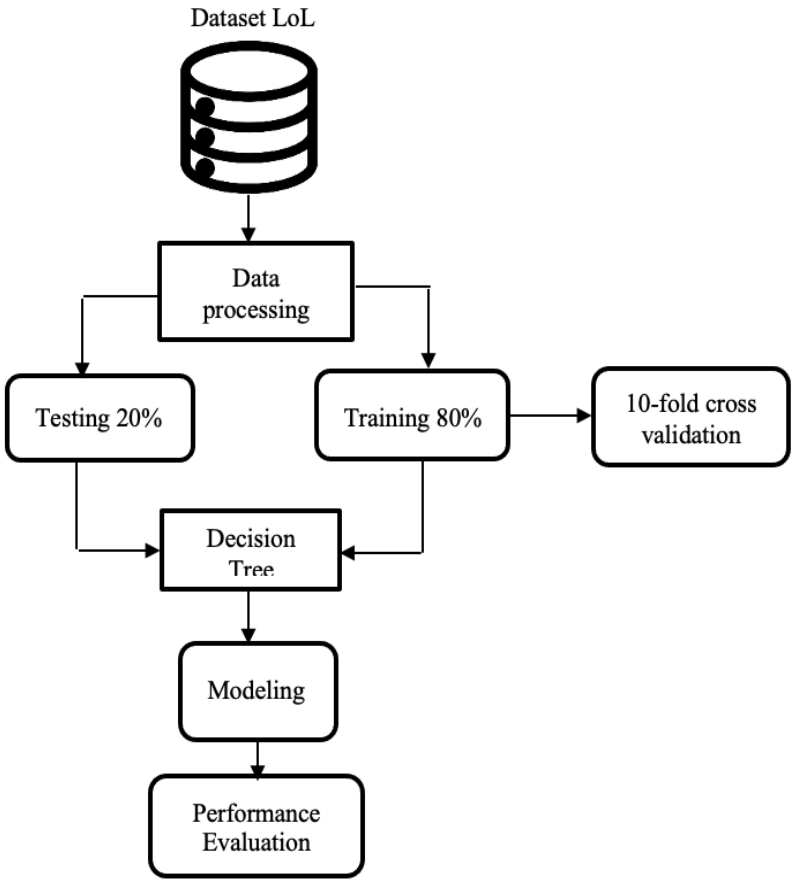
Penelitian sebelumnya oleh Putro, tentang prediksi kemenangan tim pada game *Mobile Legends* dengan menggunakan metode Naive Bayes dan hasil akurasi yang didapat sebesar 80% [7]. Kemudian pada penelitian yang lain dalam game dengan metode UCB1 untuk memenentukan pemilihan strategy dalam AI, diketahui bahwa penelitian tersebut menggunakan algoritma Decision Tree dan hasil akurasi yang didapat sebesar 84% [8]. Pada penelitian ini penulis mencoba untuk menggunakan algoritma *Decision Tree* untuk membuat model dalam menentukan kemenangan tim pada game *League of Legends*.

Tujuan dari penelitian ini adalah untuk membuat model dalam memprediksi kemenangan team dalam game *League of Legends* agar dapat membantu pemain lain dalam melihat kesempatan dalam meraih kemenangan dalam permainan. Manfaat dari penelitian ini adalah untuk membuka

pola pikir pada pemain dalam mengatur strategi dalam permainan agar dapat mengikuti pilihan yang terbaik dalam mengambil tindakan pada saat dalam permainan.

2. Metode Penelitian

Pada bagian ini menjelaskan tentang metode penelitian yang peneliti gunakan. Gambar 1 memperlihatkan proses klasifikasi game *League of Legends (LoL) ranked*. Proses pertama adalah mengambil *League of Legends (LoL) ranked dataset* yang diambil dari Kaggle. Kemudian dilakukan data preprocessing untuk mengolah data dan membersihkan data. Dataset dibagi menjadi 80% training data yang terdiri atas 40.000 data, dan 20% testing data yang terdiri atas 10.000 data. Penelitian ini menggunakan 10-fold cross validation untuk membagi data menjadi 10 bagian dan diuji sebanyak 10 kali sebelum melakukan modelling. Selanjutanya data di proses menggunakan algoritma *Decision Tree*, yang kemudian akan dievaluasi.



Gambar 1. Desain Penelitian

2.1 League of Legends Dataset

Data yang digunakan pada penelitian ini adalah *League of Legends Ranked Games (LoL) dataset* yang dapat di akses di Kaggle. Dataset ini memiliki 50000 *rows*, dan 60 attributes. Pada dataset ini terdapat beberapa parameter seperti: *gameID, creationTime, gameDuration, seasonId, winner,* jumlah *tower*, jumlah *inhibitor, baron, dragon, riftherald, champions and summoner spells, First*

Baron, dragon, tower, blood, inhibitor, Rift Herald (1 = team1, 2 = team2, 0 = none), dan 5 *hero bans* dari setiap *team*.

Tabel 1. Parameter Dataset

Parameter	Details	Value
gameId	ID user	Polynomial
creationTime	Berapa lama akun itu dibuat	Polynomial
gameDuration	Durasi dari tiap pertarungan	Polynomial
seasonId	Urutan dari setiap musim	Integer
winner	Team yang menang	Integer
firstBlood, firstTower, firstInhibitor, firstBaron, firstDragon, dan firstRiftHerald	Player yang pertama kali membunuh Player musuh, hancurkan tower, membunuh minion musuh, membunuh Baron, Dragon dan RiftHerald	Integer
t1_champ1id, t1_champ2id, t1_champ3id, t1_champ4id, dan t1_champ5id t2_champ1id, t2_champ2id, t2_champ3id, t2_champ4id, dan t2_champ5id	Urutan Champion pertama, kedua, ketiga, keempat, dan kelima Team1 dan Team 2	Polynomial
t1_champ1_sum1, t1_champ2_sum1, t1_champ3_sum1, t1_champ4_sum1, dan t1_champ5_sum1 t2_champ1_sum1, t2_champ2_sum1, t2_champ3_sum1, t2_champ4_sum1, dan t2_champ5_sum1	Urutan Champion pertama, kedua, ketiga, keempat, dan kelima Team1 dan Team 2 Sum 1	Polynomial
t1_champ1_sum2, t1_champ2_sum2, t1_champ3_sum2,	Urutan Champion pertama, kedua, ketiga, keempat, dan kelima Team1 dan Team 2 Sum2	Polynomial

t1_champ4_sum2, dan t1_champ5_sum2 t2_champ1_sum2, t2_champ2_sum2, t2_champ3_sum2, t2_champ4_sum2, dan t2_champ5_sum2		
t1_towerKills, t1_inhibitorKills, t1_baronKills, t1_dragonKills, dan t1_riftHeraldKills t2_towerKills, t2_inhibitorKills, t2_baronKills, t2_dragonKills, dan t2_riftHeraldKills	Team 1 dan Team 2 menghancurkan Tower, membunuh minion musuh, membunuh Baron, Dragon dan RiftHerald	Integer
t1_ban1, t1_ban2, t1_ban3, t1_ban4, dan t1_ban5 t2_ban1, t2_ban2, t2_ban3, t2_ban4, dan t2_ban5	Playar 1 yang diblokir Playar 2 yang diblokir	Integer

2.2 Data Preprocessing

Data preprocessing dibagi menjadi tiga bagian, yaitu: *data cleaning, data reduction, dan data transformation*. *Data cleaning* adalah proses pembersihan data *incomplete* pada *attribute* di *dataset* untuk membuat data menjadi lebih konsisten. Sedangkan, *data reduction* adalah proses untuk menghapus data pada attribute yang kurang dominan sehingga data bisa dikurangi, namun tetap menghasilkan data yang akurat. *Data transformation* adalah proses untuk mentraformasi atribut ke bentuk yang sesuai untuk pembuatan model, pada proses ini juga menormalisasi data menjadi range yang lebih kecil untuk digunakan pada proses selanjutnya [9].

Pada dataset ini, kami menemukan terlalu banyak variabel. Banyak dari atribut tersebut yang tidak mudah kita analisa dan kalkulasi. Misalnya, *"ID Champion"*, *"Champion and Summoner spells"*, dan *bans*. Jadi kami memilih untuk menghapus atribut tersebut. Kemudian, untuk data transformasi kami mengkonversi waktu menjadi format yang dapat lebih mudah dikenali. Durasi permainan juga perlu diubah, dalam hal ini diubah menjadi menit untuk analisis nanti. Semua game yang kurang dari 15 menit dihapus dari data. Bahkan permainan cepat cenderung berlangsung sekitar 20 menit, meskipun beberapa permainan profesional tercepat hanya berlangsung selama 18 menit. Permainan yang berlangsung kurang dari 18 menit kemungkinan besar telah berakhir dengan status *players disconnecting*,

2.3 Algoritma Decision Tree

Algoritma Decision Tree adalah metode pembelajaran mesin dalam membuat keputusan menggunakan model seperti pohon. Decision Tree juga bisa menyelesaikan berbagai alternatif mengatasi masalah, juga terdapat faktor-faktor yang mempengaruhi perhitungan[10]. Jarak dalam Decision Tree dihitung menggunakan Entrophy dengan rumus:

$$Entropy\ (S) = \sum_{i=1}^{n} - pi * \log_2 pi \quad \text{.....(1)}$$

Keterangan:

S : Himpunan kasus
k : Jumlah partisi *S*
Pj : Probabilitas yang didapat dari jumlah (Ya/Tidak) dibagi total kasus

2.4 Performance Measurement

Selesai dari proses pembuatan modeling maka proses selanjutnya adalah *Performance evaluation.* Proses ini berguna dalam pengujian pada *clasisifier*. *Recall* adalah proses untuk melihat dan memprediksi angka yang positif, *Precision* adalah proses dimana saat melihat dari prediksi dari total bilangan positif, *Accuracy* adalah proses saat ketika bilangan positif lebih banyak dan tepat [11]. Berikut ini adalah rumus *recall, precision, dan accuracy* dalam *performance evaluation*:

Rumus Recall:

$$\text{Recall} = \frac{TP}{TP+FN}, \text{.....(2)}$$

Rumus Precision:

$$\text{Precision} = \frac{TP}{TP+FP}, \text{..... (3)}$$

Rumus Accuracy:

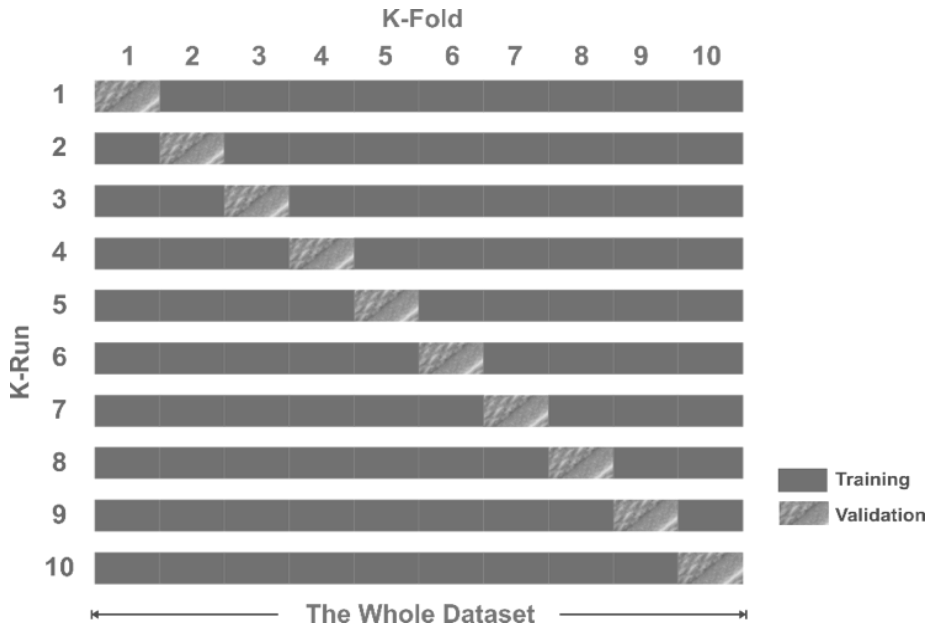
$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}, \text{.....(4)}$$

Keterangan:

TP	: Nilai true positive	FP	: Nilai false positive
TN	: Nilai true negative	N	: Jumlah data negative
P	: Jumlah data positif	FN	: Nilai false negative

2.4.1 10-fold Cross Validation

Cross Validation adalah teknik untuk mengevaluasi model pada data *training* dengan cara mempartisi sampel asli ke dalam *training data* untuk melatih model, dan *testing data* untuk mengevaluasi model. Dalam *k-fold cross validation*, sampel asli secara acak dipartisi dalam *k equal size subsample*. Dari *subsample k*, satu *subsample* akan digunakan sebagai *testing data* dan sisanya akan menjadi *training data*.



Gambar 2. ***10-fold Cross Validation***

2.5 Feature Selection

Feature selection atau yang biasanya disebut juga *feature reduction* dengan tujuan untuk memilih feature yang bagus dan yang tidak bagus disampingkan pada suatu kegiatan pemodelan atau penganalisaan data. Banyak sekali cara yang bisa digunakan dan harus dilakukan berulang-ulang untuk mencari yang cocok[12]. Penulis menggunakan *information gain(ig)* dari suatu term diukur dengan menghitung jumlah bit informasi yang di ambil dari prediksi kategori dengan ada atau tidaknya term pada suatu data [13]. secara matematis rumus *information gain* sebagai berikut:

$$InfoGain\ (\mathrm{S,A}) = \mathrm{Entropy(S)} - \sum\left(v\ \in Value(A)\right)\frac{|\mathrm{S_v}|}{\mathrm{S}}\ Entropy\ (\mathrm{S\_v}) \cdots\cdots..(5)$$

Keterangan:

S	: Jumlah seluruh fitur
A	: kategori
S_v	: jumlah sampel untuk nilai v
v	: nilai yang mungkin untuk kategori A
S_i	: fitur ke i
value(A)	: himpunan nilai-nilai yang mungkin untuk kategori A

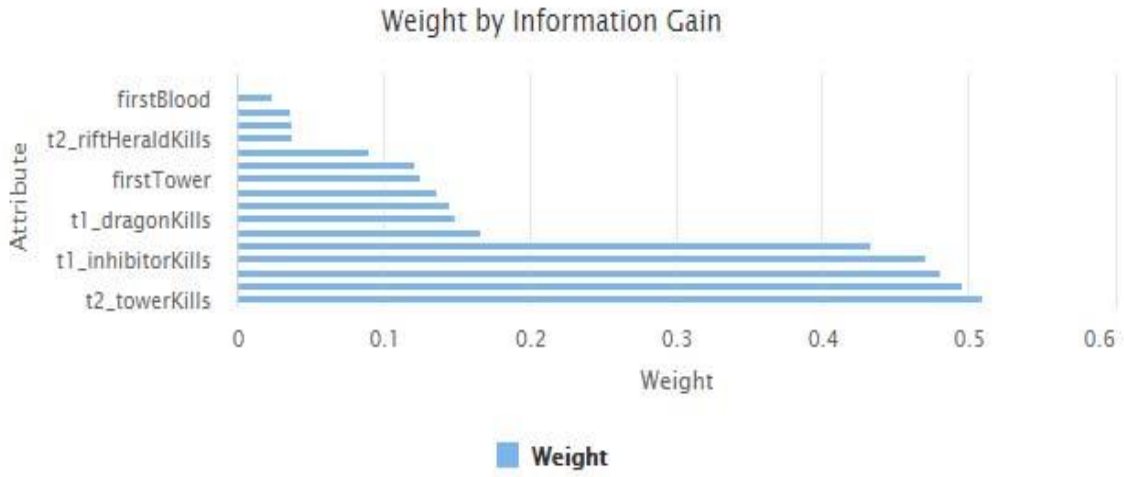
3. Hasil dan Pembahasan

Pada bagian ini penulis melakukan analisa terhadap *(LoL) League of Legends Ranked Games* dataset menggunakan algoritma *Decition Tree*.

3.1 Feature Selection Menggunakan Information Gain

Peneliti menggunakan *information gain* untuk mengetahui fitur yang memiliki informasi penting dalam pembuatan model. Berdasarkan Gambar 3 dapat disimpulkan bahwa bahwa attribute *t2_towerKills* dan *t1_inhibotorKills* memiliki *weight* sebesar *0.5*. Untuk attribute lain

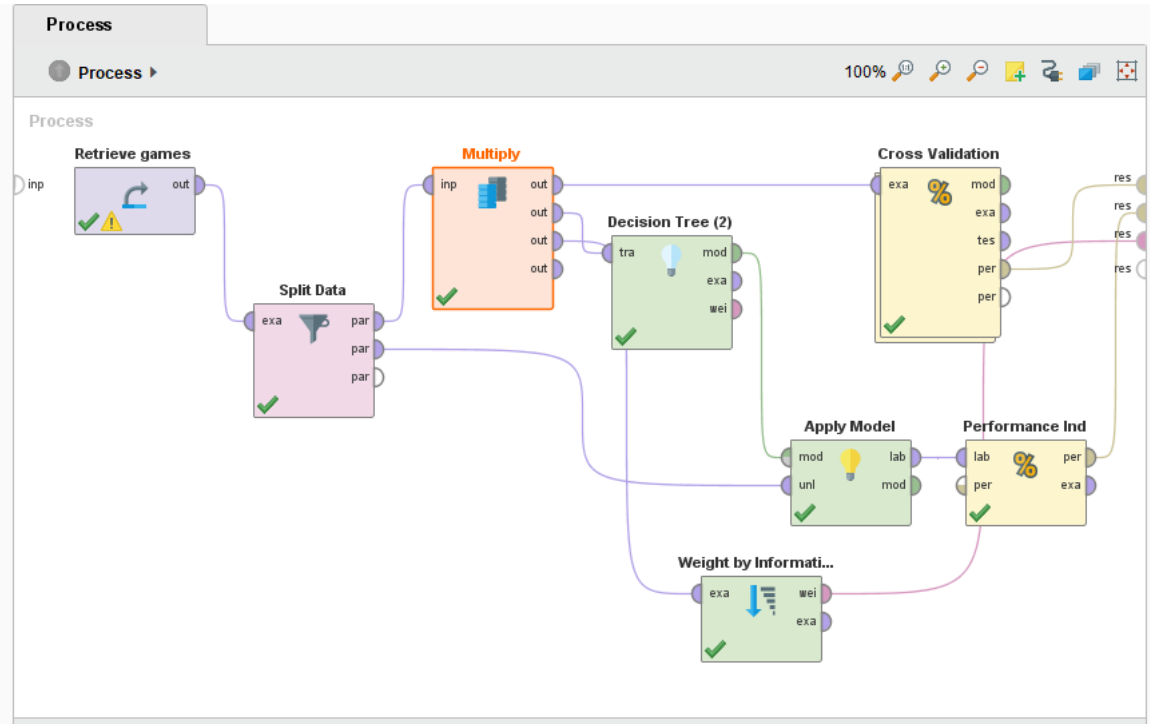
t1_dragonKills 0.2, firstTower 0.15, t2_riftHeraldKills dan *firstBlood* dengan *Weight* sebesar *0.05*. *Weight* dari *information gain* untuk setiap atribut digunakan untuk memilih fitur atau atribut yang memiliki informasi yang penting yang selanjutnya digunakan pada proses pembuatan tree menggunakan *decision tree.*



Gambar 3. Feature importance dengan Information Gain

3.2 Use Case Pembuatan Model pada Rapidminer

Pada tahap ini peneliti mendesain *use case* proses pembuatan model dengan menggunakan *Software Rapidminer*. Gambar 4 adalah Langkah pembuatan model: pertama masukan dataset ke *design proses*, kemudian data dibagi menjadi data training 80% dan data testing 20%. Data training mempunyai 2 proses yaitu *Cross Validation* dan pengujian *Independent*, pada proses *cross validation*, data training dibagi menjadi *10-fold cross validation* dan setiap subset akan dilatih menggunakan algoritma *Decision Tree* dan dilihat perfoma dari model yang dibuat, proses yang kedua adalah pengujian *Independent* data atau data testing dimana pada proses ini data dilatih menggunakan algoritma *Decision Tree* selanjutnya menguji model yang dibuat dengan menggunakan *confusion matrix.*



Gambar 4. Use Case Proses Pembuatan Model

3.3 Komparasi Algoritma Menggunakan Independent Dataset

Pada Tabel 2 memperlihatkan hasil performance evaluation dari algoritma klasifikasi tanpa menggunakan *cross validation* yaitu dengan independent dataset. Dari hasil tersebut dapat dilihat bahwa algoritma *decision tree* memiliki *accuracy*, *recall*, *precision*, dan *RMSE* tertinggi dibandingkan dengan algortima lain dengan nilai *accuracy* sebesar 96.42 %, *recall* Team 1 sebesar 97.74%, *recall* Team 2 sebesar 95.06%, *precision* Team 1 sebesar 95.31%, *precision* Team 2 sebesar 97.62% dan nilai *RMSE* 0.157.

Tabel 2. Komparasi Algoritma Menggunakan Independent Dataset

Independent Dataset						
Algoritma	**Accuracy (%)**	**Recall (%)**		**Precision (%)**		**RMSE**
		Team 1	Team 2	Team 1	Team 2	
Decision Tree	96.42	97.74	95.06	95.31	97.62	0.157
Logistic Regression	96.39	96.59	96.18	96.29	96.49	0.159
Naïve Bayes	93.72	94.63	92.78	93.08	94.40	0.238

Berdasarkan Table 2 maka algoritma decision tree memiliki performa yang baik dibanding dengan algoritma logistic regression dan naïve bayes, dilihat dari nilai akurasi, recall, precision, dan RMSE.

3.4 Komparasi Algoritma Menggunakan 10-fold Cross Validation

Tabel 3 menunjukkan hasil *performance evaluation* dari algoritma klasifikasi yang menggunakan *10-fold cross validation*. Dari hasil tersebut diketahui bahwa algoritma *Decision Tree* memiliki accuracy, dan recall tertinggi dibandingkan dengan algoritma lain dengan nilai accuracy sebesar 96.24%, recall sebesar 97.34% untuk Team 1, 95.11% recall untuk Team 2, kemudian nilai precision sebesar 95.33% untuk Team 1 dan, 97.21% untuk Team 2 serta nilai RMSE sebesar 0.161. Algoritma *Logistic Regression* memiliki tingkat precision dan RMSE tertinggi ke dua dibandingkan dengan *Naive Bayes* yang memiliki selisih sebesar team 1 = 3.01%, team 2 = 1.27% precision dan 0.007 RMSE.

Tabel 3. Komparasi Algoritma Menggunaakan Independent Dataset

Hasil 10 Folds Cross Validation						
Algoritma	**Accuracy (%)**	**Recall (%)**		**Precision (%)**		**RMSE**
		Team 1	Team 2	Team 1	Team 2	
Decision Tree	96.24	97.34	95.11	95.33	97.21	0.161
Logistic Regression	96.04	96.03	96.04	96.14	95.93	0.165
Naïve Bayes	93.87	94.90	92.81	93.13	94.66	0.235

4. Kesimpulan

Dari tiga algoritma yang telah di evaluasi yaitu *Decision Tree, Logistic Regression,* dan *Naïve Bayes* dapat disimpulkan bahwa algoritma *Decision Tree* memiliki performa yang paling baik di antara algoritma lainnya dengan hasil 96.42% *accuracy*, 97.74% *recall Team 1*, 95.06% *recall Team 2*, 95.31% *precision Team 1*, 97.62% *precision Team 2* dan 0.157 RMSE untuk hasil independent sedangkan untuk hasil *10-fold cross validation* memiliki hasil 96.24% *accuracy*, 97.34 % *recall Team 1*, 95.11% *recall Team 2*, 95.33% *precision Team 1*, 97.21% *precision Team 2*, dan 0.161 RMSE dalam mendeteksi kemenangan pada game *League of Legend*. Untuk kedepannya diharapkan model ini dapat berguna untuk meningkatkan kualitas pemain dalam game ini dan untuk peneliti selanjutnya diharapkan peniliti dapat menggunakan metode dan algoritma yang lain agar dapat memaksimalkan hasil dan mengurangi error dalam pemodelan.

Daftar Pustaka

[1] Kompas Media, "Perkembangan Teknologi Informasi dan Komunikasi di Indonesia Halaman all - Kompas.com", KOMPAS.com, 2021. [Online]. Available: https://www.kompas.com/skola/read/2020/12/21/164007469/perkembangan-teknologi-informasi-dan-komunikasi-di-indonesia?page=all.

[2] "Melihat Tren Perkembangan Smartphone | merdeka.com", merdeka.com, 2021. [Online]. Available: https://www.merdeka.com/teknologi/melihat-tren-perkembangan-smartphone.html.

[3] S. Aggarwal, S. Saluja, V. Gambhir, S. Gupta and S. Satia, "Predicting likelihood of psychological disorders in PlayerUnknown's Battlegrounds (PUBG) players from Asian countries using supervised machine learning", Addictive Behaviors, vol. 101, p. 106132, 2020. Available: 10.1016/j.addbeh.2019.106132.

[4] S. Demediuk et al., "Player retention in league of legends", Proceedings of the Australasian Computer Science Week Multiconference, 2018. Available: 10.1145/3167918.3167937.

[5] X. Zhang, Y. Yue, X. Gu, B. Niu and Y. Feng, "Investigating the Impact of Champion Features and Player Information on Champion Usage in League of Legends", Proceedings of the 2017 International Conference on Information Technology - ICIT 2017, 2017. Available: 10.1145/3176653.3176730.

[6] X. Sheng and D. Thuente, "Using Decision Trees for State Evaluation in General Game Playing", KI - Künstliche Intelligenz, vol. 25, no. 1, pp. 53-56, 2011. Available: 10.1007/s13218-010-0079-2.

[7] A.C.Putro, "SISTEM PREDIKSI KEMENANGAN TIM PADA GAME MOBILE LEGENDS DENGAN METODE NAIVE BAYES," *Teknik Informatika*, vol. 1, p. 11, 2018.

[8] G. R. E. Santoso, "Pembuatan Game dengan Menerapkan Metode Decision Tree: UCB1, untuk Menentukan Pemilihan Strategy dalam AI.," *Jurnal Infra*, vol. 1, p. 7, 2017.

[9] G. A. Sandag, N. E. Tedry and S. Lolong, "Classification of Lower Back Pain Using K-Nearest Neighbor Algorithm," 2018 6th International Conference on Cyber and IT Service Management (CITSM), Parapat, Indonesia, 2018, pp. 1-5, doi: 10.1109/CITSM.2018.8674361.

[10] U. Arni, "Pengertian dan Penerapan Decision Tree," garudacyber.co.id, 15 January 2020. [Online]. Available: https://garudacyber.co.id/artikel/1545-pengertian-dan-penerapan-decision-tree. [Accessed 19 November 2019].

[11] G. A. Sandag, F. Gara and Irfan, "Android Application Market Prediction Based on User Ratings Using KNN," 2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS), Manado, Indonesia, 2020, pp. 1-5, doi: 10.1109/ICORIS50180.2020.9320812.

[12] A. T. Liem, G. A. Sandag, I-S. Hwang, and A. Nikoukar, "Delay Analysis of Dynamic Bandwidth Allocation for Triple-Play-Services in EPON," p. 6, 2017.

[13] G. Sandag, J. Leopold and V. Ong, "Klasifikasi Malicious Websites Menggunakan Algoritma K-NN Berdasarkan Application Layers dan Network Characteristics", CogITo Smart Journal, vol. 4, no. 1, p. 37, 2018. Available: 10.31154/cogito.v4i1.100.37-45.