

# PREDIKSI PERILAKU PEMAIN MINECRAFT MENGGUNAKAN PLUGIN SPIGOT DENGAN ALGORITMA DECISION TREE DAN RANDOM FOREST

Andrew Immanuel<sup>1</sup>

Email: <sup>1</sup>graymontstudio@gmail.com

## Abstrak

Minecraft adalah permainan *sandbox* yang memberikan kebebasan bagi pemain untuk melakukan berbagai aktivitas seperti menambang, membangun, dan berburu. Setiap pemain menunjukkan pola perilaku yang berbeda-beda. Penelitian ini bertujuan untuk mengklasifikasikan perilaku pemain Minecraft berdasarkan data aktivitas in-game yang dikumpulkan menggunakan plugin Spigot khusus bernama *PlayerBehaviour*. Data yang dicatat meliputi jumlah blok dihancurkan, blok dipasang, mob dikalahkan, item dikraft, jarak tempuh, dan berbagai aktivitas lainnya. Data kemudian dilabeli secara otomatis berdasarkan dominasi aktivitas tertentu, dan digunakan untuk melatih model klasifikasi dengan algoritma *Decision Tree* dan *Random Forest*. Hasil pelatihan menunjukkan bahwa algoritma *Random Forest* memberikan akurasi terbaik sebesar **74%**, mengungguli *Decision Tree* yang hanya mencapai **39%**. Model ini berpotensi digunakan untuk personalisasi pengalaman bermain serta mendeteksi perilaku tidak wajar pemain.

**Kata kunci:** Minecraft, Perilaku Pemain, Plugin Spigot, Data Aktivitas, Klasifikasi, Random Forest

## ***PREDICTION OF MINECRAFT PLAYER BEHAVIOR USING SPIGOT PLUGIN WITH DECISION TREE AND RANDOM FOREST ALGORITHM***

## Abstract

*Minecraft is a sandbox game that allows players to engage in a wide range of activities such as mining, building, fighting, and hunting. Each player exhibits distinct behavioral patterns. This study aims to classify player behavior based on in-game activity data collected through a custom Spigot plugin named PlayerBehaviour. The recorded data includes blocks mined, blocks placed, mobs killed, items crafted, distance traveled, and other activities. Labels were automatically assigned based on dominant activity patterns. Classification models were trained using Decision Tree and Random Forest algorithms. The results show that Random Forest achieved the highest accuracy of 74%, outperforming the Decision Tree with only 39%. This model has potential applications in gameplay personalization and abnormal behavior detection.*

**Keywords:** *Minecraft, Player Behavior, Spigot Plugin, Activity Data, Classification, Random Forest*

---

## 1. PENDAHULUAN

Minecraft merupakan salah satu permainan *sandbox* yang paling populer di dunia, dengan lebih dari 140 juta pemain aktif bulanan dari berbagai kalangan usia. Dalam permainan ini, pemain diberi kebebasan untuk melakukan berbagai aktivitas seperti menambang, membangun, menjelajah, bertarung, hingga bercocok tanam. Keberagaman aktivitas ini menciptakan pola perilaku yang unik dari setiap pemain.

Pemahaman terhadap perilaku pemain dalam permainan daring seperti Minecraft memiliki potensi yang besar, baik dalam konteks pengembangan sistem adaptif, rekomendasi fitur dalam gim, maupun sebagai bagian dari penelitian dalam bidang analisis data perilaku digital. Sayangnya, pengklasifikasian perilaku pemain secara otomatis berdasarkan aktivitas dalam gim masih belum banyak dikembangkan, khususnya pada permainan

seperti Minecraft yang bersifat terbuka dan tidak memiliki jalur permainan yang linear.

Seiring berkembangnya teknologi *plugin* seperti Spigot, pengembang dapat merekam data aktivitas pemain secara real-time langsung dari server Minecraft. Data aktivitas tersebut dapat dianalisis lebih lanjut menggunakan algoritma machine learning untuk mengidentifikasi pola perilaku pemain.

Dalam penelitian ini, dilakukan pengembangan *plugin* Minecraft bernama PlayerBehaviour yang mampu mencatat aktivitas pemain seperti jumlah blok yang dihancurkan, jumlah mob yang dikalahkan, jarak yang ditempuh, dan lain-lain. Data tersebut kemudian digunakan untuk membangun model klasifikasi perilaku pemain menggunakan algoritma Decision Tree, yang bertujuan untuk memprediksi tipe perilaku seperti *miner*, *builder*, *hunter*, dll.

Penelitian ini diharapkan dapat memberikan kontribusi dalam pengembangan sistem analisis perilaku pemain secara otomatis serta membuka peluang penerapan lebih lanjut dalam bidang *game analytics*, personalisasi pengalaman bermain, dan deteksi pemain dengan perilaku abnormal.

## 2. KAJIAN LITERATUR

### 2.1 What is player behavior?

*Player behavior can be complicated. That's because it's a combination of what our students, players, and users are experiencing and how they react to that experience. Player behavior is influenced by human behavior which is a complex, reflective, and impulsive process (Eng, 2020).*

*However, for the purposes of games-based learning; serious games; and simulations we consider player behavior the way that the players acts and conducts themselves within the game. This means that player-to-player interactions; actions within the game; and even responses to the environment are taken into content within player behavior. Because of this, player behavior often begins at the cusp of the "magic circle" and the*

*ludological agreement of gaming. That's where the rules of the game begin and the rules of the real world end. Crossing this border signals that the player is now within a different world that has different expectations within the game (Eng, 2020).*

*Crossing this boundary is what we do whenever we plays games. But crossing that boundary could be dynamic for players. In addition, it's also susceptible to when and for how long players engage; the location (physically) where they play; the situation and social connections of who they play with; as well as other characteristics that make up the context of their experience (Eng, 2020).*

### 2.2 Decision Trees

*A decision tree is a non-parametric supervised learning algorithm. It has a hierarchical, tree structure, which consists of a root node, branches, internal nodes and leaf nodes (Koli, 2023).*

*Decision Trees are the foundation for many classical machine learning algorithms like Random Forests, Bagging, and Boosted Decision Trees. His idea was to represent data as a tree where each internal node denotes a test on an attribute (basically a condition), each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label (Koli, 2023).*

### 2.3 Random Forest

*A Random Forest is an ensemble machine learning model that combines multiple decision trees. Each tree in the forest is trained on a random sample of the data (bootstrap sampling) and considers only a random subset of features when making splits (feature randomization).*

*For classification tasks, the forest predicts by majority voting among trees, while for regression tasks, it averages the predictions. The model's strength comes from its "wisdom of crowds" approach – while individual trees might make errors, the collective decision-making process tends to average out these*

*mistakes and arrive at more reliable predictions (Baladram, 2024).*

### 3. METODE

#### 3.1 Desain Penelitian

Penelitian ini menggunakan pendekatan eksperimen komputasi dengan metode klasifikasi *supervised learning*. Tujuan utama penelitian adalah membangun model klasifikasi yang mampu memprediksi perilaku pemain Minecraft berdasarkan data aktivitas *in-game* yang direkam melalui plugin Spigot.

Desain penelitian terdiri dari beberapa tahap utama, yaitu pengumpulan data aktivitas pemain, pra-pemrosesan data, pelabelan kelas perilaku, pelatihan model klasifikasi menggunakan algoritma *Decision Tree* dan *Random Forest*, serta evaluasi performa model dengan metrik akurasi, *precision*, *recall*, dan *f1-score*. Penelitian ini bersifat kuantitatif dengan fokus pada analisis performa algoritma dalam mengklasifikasikan perilaku pemain berdasarkan data aktivitas yang bersifat numerik.

#### 3.2 Pengumpulan Data

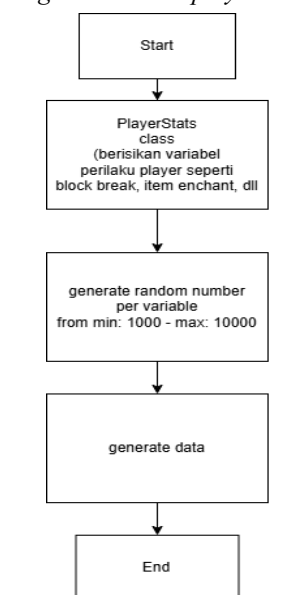
Data aktivitas pemain Minecraft dikumpulkan menggunakan *plugin* Spigot khusus bernama *PlayerBehaviour* yang dikembangkan untuk penelitian ini. Plugin ini diinstal pada *server Minecraft* dan secara otomatis mencatat berbagai aktivitas pemain dalam permainan. Aktivitas yang dicatat meliputi jumlah blok yang dihancurkan, jumlah blok yang dipasang, jumlah mobs yang dikalahkan, jumlah item yang dibuat, jarak tempuh, jumlah kayu yang ditebang, jumlah hewan yang dikembangbiakkan, jumlah item yang *dienchant*, jumlah ikan yang ditangkap, dan jumlah *potion* yang dibuat. Setiap entri data terkait pemain disimpan secara terstruktur dalam format CSV, dengan tambahan kolom pengenalan seperti UUID dan nama pemain. Data dikumpulkan dari aktivitas pemain di server dalam kurun waktu tertentu untuk memastikan variasi perilaku yang cukup.

Dalam plugin *PlayerBehaviour* terdapat fungsi untuk melakukan *generate data* secara otomatis yang bervariasi. Data-data hasil *generate* akan menghasilkan data player buatan secara *random*. Dalam pengumpulan data, secara umum bisa dilakukan dengan memasang plugin di salah satu *server* dengan versi 1.21.4 saat ini untuk mendapatkan data dari pemain asli, namun untuk penelitian kali ini data yang diperoleh dari hasil *generate data*.

Dalam proses *generate data*, terdapat juga proses labeling untuk memberikan label pada setiap *player*. Pelabelan ini bertujuan untuk memberikan tanda perilaku player pada setiap data. *Labeling* ini akan melihat sang player lebih condong kearah mana.

Proses *labeling* akan dilakukan secara otomatis dengan melihat *value* tertinggi dari setiap variabel yang sudah di generate oleh function dari plugin *PlayerBehaviour*. *Value* tertinggi yang didapat akan menentukan label yang didapat pada *player* tersebut.

Berikut adalah langkah-langkah untuk melakukan *generate data player*:



Flowchart 1. Pengumpulan Data

### 3.3 Data Cleansing

Data yang terkumpul melalui plugin kemudian dilakukan pembersihan untuk menyiapkan dataset yang bersih dan terstruktur. Langkah pembersihan meliputi penghapusan kolom non-fitur seperti UUID dan nama pemain yang tidak relevan untuk klasifikasi, penanganan nilai kosong jika ditemukan, serta pemisahan dataset menjadi data latih dan data uji dengan rasio 80 banding 20 menggunakan teknik train-test split. Pra-pemrosesan bertujuan memastikan data siap digunakan dalam pelatihan model klasifikasi dengan representasi fitur yang sesuai.

label	blocks_mined	crops_harvested	logs_broken	fish_caught	meats_killed	blocks_placed	items_enchanted	items_crafted	portions_brewed	animals_bred
farmer	7997	9539	2390	1030	1721	8871	9088	3146	6017	6801
miner	1429	9057	7523	8540	6367	6387	3470	2762	6471	3882
builder	2410	4905	9218	4650	2156	4956	3599	7266	4502	8872
miner	8717	1056	8485	8017	1506	2900	6139	8294	1471	7853
lumberjack	3516	2920	9999	3635	2506	5774	9380	2212	7379	9617
hunter	1335	4618	6642	8797	9628	9530	2788	6544	7754	2054
craftsman	6792	5955	3799	3941	1792	8084	4304	9462	5321	7100
builder	4869	5093	5351	6219	7872	8217	6562	1240	6203	2574
fisherman	5500	5689	6817	9504	7603	4748	6624	3664	1554	2141
hunter	2346	1634	8614	4033	9058	3537	5743	8192	1461	2435
fisherman	8495	2254	4041	9240	2624	5139	4535	8279	2888	1628
craftsman	3057	8134	9511	9262	9231	8873	2444	9844	7886	4904
miner	8345	2303	3558	2851	1854	6075	7943	7635	5002	8510
fisherman	9180	8964	2916	9589	8608	8056	2993	6425	4191	2683
farmer	4336	9658	8622	4186	3853	8008	5407	7799	8061	5488
miner	7453	5381	6600	4296	1719	4487	5559	4870	1751	9198
builder	3314	7461	5475	9285	6838	9180	6205	2461	1521	8197
animal breeder	3446	1781	5085	4837	6019	3047	7446	2377	7811	8696
animal breeder	6296	1561	7983	8647	9114	4780	1186	1046	1044	8851
animal breeder	6916	9053	1074	1604	1888	6259	4047	2479	8899	9508
builder	1076	4884	1005	1248	7623	9923	1411	8835	1070	9048
miner	8913	2078	4862	1172	2872	8127	4408	7387	7770	1181
miner	9410	1586	1704	1775	2694	1740	1602	6286	4464	4311
fisherman	7583	1156	1065	9185	7948	1289	7007	4753	5239	3178
lumberjack	2622	6186	9750	5430	1705	3006	3448	8136	1108	1848
lumberjack	1644	6430	9637	1504	8024	5484	2428	6797	8212	8293

Tabel 1. *Cleaned Data*

Pada tabel 1 diperlihatkan bahwa kolom pada dataset tersisa label dan variabel-variabel lainnya seperti blocks\_mined, crops\_harvested, logs\_broken, dll.

### 3.4 Model Training

Pada tahap ini, kedua algoritma yaitu *Decision Tree* dan *Random Forest* akan dilakukan *training* menggunakan *dataset*. Hasil akurasi yang didapat akan menentukan algoritma mana yang akan dijadikan model dan digunakan untuk prediksi perilaku player.

Training dataset dilakukan menggunakan tools berupa script program dengan bahasa pemrograman python. Script ini terdiri dari 2 *script* yang berbeda. Script pertama akan memuat script untuk melatih model *decision tree* dan *script* kedua untuk melatih model *random forest*.

#### a. Decision Tree Training

	precision	recall	f1-score	support
alchemist	0.00	0.00	0.00	12
animal breeder	0.00	0.00	0.00	20
builder	0.62	0.28	0.38	18
craftsman	0.15	0.82	0.25	17
enchanter	0.00	0.00	0.00	23
farmer	0.52	0.70	0.59	23
fisherman	0.69	0.45	0.55	20
hunter	0.00	0.00	0.00	19
lumberjack	0.62	0.58	0.60	26
miner	0.63	0.86	0.73	22
accuracy			0.39	200
macro avg	0.32	0.37	0.31	200
weighted avg	0.35	0.39	0.34	200

Gambar 1. *Decision Tree Model Training Result*

Setelah proses *training* pertama, akurasi *decision tree* hanya sebesar 0.39 (39%) yang menandakan bahwa algoritma ini kurang akurat untuk melakukan prediksi perilaku *player*.

#### b. Random Forest

	precision	recall	f1-score	support
alchemist	0.75	0.75	0.75	12
animal breeder	1.00	0.60	0.75	20
builder	0.80	0.44	0.57	18
craftsman	0.88	0.82	0.85	17
enchanter	0.87	0.87	0.87	23
farmer	0.61	0.74	0.67	23
fisherman	0.71	0.85	0.77	20
hunter	0.85	0.58	0.69	19
lumberjack	0.74	0.77	0.75	26
miner	0.60	0.95	0.74	22
accuracy			0.74	200
macro avg	0.78	0.74	0.74	200
weighted avg	0.77	0.74	0.74	200

Gambar 2. *Random Forest Model Training Result*

Setelah proses *training* kedua, akurasi *random forest* cukup besar yaitu 0.74 (74%) yang menandakan bahwa algoritma ini cukup akurat untuk melakukan prediksi perilaku *player* *minecraft*.

### 3.5 Saving Model

Dalam program *python*, setelah melakukan *training*, model *random forest* akan disimpan kedalam file berformat *.pkl*. Model yang dihasilkan ini nantinya akan digunakan untuk melakukan prediksi perilaku *player*.

```
joblib.dump(rf_clf, 'player_behaviour_model.pkl')
```

Gambar 3. *Player Behaviour Model*

### 3.6 Model Testing

Model yang sudah ada akan dilakukan *testing* dengan variabel testing untuk melakukan prediksi *player* minecraft. Proses testing dilakukan menggunakan *script python*.

```
import joblib
import pandas as pd

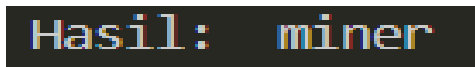
# Load model
loaded_model = joblib.load('player_behaviour_model.pkl')

# Data Player
new_player_data = pd.DataFrame([
    'blocks_mined': 50000,
    'crops_harvested': 2000,
    'logs_broken': 100,
    'fish_caught': 50,
    'mobs_killed': 8000,
    'blocks_placed': 1000,
    'items_enchanted': 5,
    'items_crafted': 20,
    'potions_brewed': 10,
    'animals_bred': 10
])

predicted_label = loaded_model.predict(new_player_data)
print("Hasil: ", predicted_label[0])
```

Gambar 4. Model Testing Script

New\_player\_data memuat data dari setiap variabel dari dataset. Keseluruhan data merepresentasikan satu player yang akan di testing menggunakan model *random forest* yang sudah disimpan.



Gambar 5. Model Testing Result

Hasil menunjukkan bahwa sang player masuk kedalam kategori *miner* dengan *blocks mined* sebanyak 50000 *blocks*.

## 4. PEMBAHASAN

### 4.1 Analisis Hasil Model Klasifikasi

Berdasarkan hasil pelatihan model, terlihat bahwa algoritma *Random Forest* memiliki performa yang jauh lebih baik dibandingkan *Decision Tree* dalam mengklasifikasikan perilaku pemain Minecraft. Model *Random Forest* mencapai akurasi 0.74 (74%), sedangkan *Decision Tree* hanya menghasilkan akurasi 0.39 (39%).

Perbedaan performa ini dapat dijelaskan dari karakteristik masing-masing algoritma. *Decision Tree* cenderung mudah *overfitting* terhadap data pelatihan, terutama jika dataset memiliki banyak variabel dan kompleksitas tinggi. Sebaliknya, *Random Forest* menggunakan pendekatan ensemble learning dengan menggabungkan banyak pohon keputusan, sehingga lebih stabil dan akurat

karena dapat menangani variabilitas antar data dengan lebih baik.

### 4.2 Potensi Penggunaan Model

Model ini berpotensi digunakan dalam berbagai konteks di dalam server Minecraft, seperti:

- Memberikan reward atau misi yang sesuai dengan kecenderungan perilaku pemain.
- Mengelompokkan pemain untuk event berbasis peran, seperti lomba antar *miner* atau *crafter*.
- Memberikan Rekomendasi tutorial atau tips otomatis untuk pemain baru berdasarkan gaya bermainnya.
- Memberikan ide dan referensi update terbaru untuk pengembangan server sesuai dengan minat berdasarkan perilaku player di server.

Namun, untuk penggunaan di server nyata, model perlu dilatih ulang menggunakan data aktual dari pemain asli agar hasilnya lebih akurat dan representatif.

## 5. KESIMPULAN

Penelitian ini berhasil membangun model klasifikasi perilaku pemain Minecraft menggunakan pendekatan *supervised learning* dengan algoritma *Decision Tree* dan *Random Forest*. Data aktivitas pemain dikumpulkan melalui plugin *PlayerBehaviour* yang dikembangkan secara khusus untuk mencatat berbagai jenis interaksi pemain di dalam game.

Berdasarkan hasil pelatihan dan evaluasi model, algoritma *Random Forest* menunjukkan performa yang paling optimal dengan akurasi sebesar 74%, dibandingkan dengan *Decision Tree* yang hanya mencapai akurasi 39%. Hal ini menunjukkan bahwa *Random Forest* lebih efektif dalam mengklasifikasikan data aktivitas pemain yang memiliki banyak variabel numerik.

Proses labeling otomatis berdasarkan aktivitas dominan memungkinkan klasifikasi

awal terhadap perilaku pemain, Model yang dikembangkan memiliki potensi besar untuk diterapkan dalam sistem personalisasi di dalam game, seperti penyesuaian misi, *reward*, atau pengelompokan pemain berdasarkan perilaku dominan.

Saran untuk penelitian selanjutnya adalah mengumpulkan data dari pemain nyata di server publik untuk meningkatkan validitas model, serta mempertimbangkan penggunaan metode lain seperti *K-Nearest Neighbors*, SVM, atau teknik *deep learning* untuk mendapatkan performa klasifikasi yang lebih tinggi dan lebih adaptif terhadap variasi data.

#### DAFTAR PUSTAKA

Eng, D. (2020, June 19).

*What is player behavior?*

Medium.

(<https://medium.com/@davengdesign/what-is-player-behavior-cdaa07271f86>)

Koli, S. (2023, February 27).

*Decision Trees: A Complete Introduction With Examples.*

Medium.

(<https://medium.com/@MrBam44/decision-trees-91f61a42c724>)

Baladram, S. (2024, November 7).

*Random Forest, Explained: A Visual Guide with Code Examples.*

Towards Data Science.

(<https://towardsdatascience.com/random-forest-explained-a-visual-guide-with-code-examples-9f736a6e1b3c/>)