# PENERAPAN ARTIFICIAL INTELLIGENCE UNTUK PREDIKSI PENGGUNAAN RESOURCE CPU DAN RAM PADA SERVER DEBIAN 12 MENGGUNAKAN METODE LINEAR REGRESSION

**Cantriya Anastasya Simbolon[1], Paulina Gorat[2], Naomi Simatupang[3], Lotar Mateus Sinaga[4]**

[1,2,3,4]*Universitas Katolik Santo Thomas, Medan*
E-mail : [1]cantriyasimbolonbelajar@gmail.com, [2]paulinagorat3@email.com, [3]naomisimatupang058@gmail.com, [4]lotarmateus88@gmail.com

## ABSTRAK

Pengelolaan sumber daya (*resource*) server yang optimal merupakan aspek krusial dalam menjaga stabilitas dan ketersediaan layanan sistem informasi. Keterbatasan pemantauan yang bersifat reaktif seringkali membuat administrator terlambat menangani lonjakan penggunaan CPU dan RAM yang tidak terduga, yang berpotensi menyebabkan penurunan performa hingga kegagalan sistem (*downtime*). Oleh karena itu, penelitian ini bertujuan untuk merancang, mengimplementasikan, dan menguji sistem kecerdasan buatan proaktif untuk memprediksi tingkat penggunaan *resource* CPU dan RAM di masa mendatang pada lingkungan server bersistem operasi. Linux Debian 12. Penelitian ini menggunakan pendekatan pemodelan *Machine Learning* dengan algoritma *Linear Regression*. Alur penelitian diawali dengan otomasi pengumpulan data menggunakan *Bash Script* yang mengeksekusi utilitas Linux (vmstat dan free) setiap 60 detik untuk merekam metrik secara *real-time* ke dalam format CSV. Selanjutnya, dilakukan pelatihan model AI menggunakan Python dengan pustaka Pandas untuk ekstraksi data dan Scikit-Learn untuk melatih model regresi. Variabel independen yang digunakan adalah indeks waktu historis, sedangkan variabel dependennya adalah persentase memori dan prosesor. Model yang terbentuk diekspor menggunakan modul Pickle untuk fungsi prediksi lanjutan. Hasil implementasi pada server Debian 12 (Bookworm) menunjukkan bahwa skrip *data logger* berjalan stabil dan model AI berhasil dilatih secara dinamis menggunakan data historis tersebut. Pada fase prediksi, program mampu memuat ulang model yang tersimpan dan menampilkan estimasi persentase beban CPU dan RAM untuk interval 1, 5, dan 10 menit ke depan dengan mengkalkulasi tren data historis secara akurat. Sistem ini terbukti fungsional sebagai alat bantu peringatan dini (*early warning tool*) yang ringan untuk manajemen infrastruktur server.

**Kata Kunci :** *Kecerdasan Buatan, Regresi Linear, Prediksi Beban Server, Linux Debian, Pemantauan Proaktif*

**ABSTRACT**

Optimal server resource management is a crucial aspect of maintaining the stability and availability of information system services. The limitations of reactive monitoring often cause administrators to be too late in addressing unexpected spikes in CPU and RAM usage, which can potentially lead to performance degradation or system failure (downtime). Therefore, this study aims to design, implement, and test a proactive artificial intelligence system to predict future CPU and RAM resource usage levels in a server environment running the Debian 12 Linux operating system. This study employs a machine learning modeling approach using the linear regression algorithm. The research process begins with automated data collection using a Bash script that executes Linux utilities (vmstat and free) every 60 seconds to record metrics in real time in CSV format. Next, the AI model is trained using Python with the Pandas library for data extraction and Scikit-Learn for training the regression model. The independent variable used is the historical time index, while the dependent variables are the memory and CPU usage percentages. The resulting model is exported using the Pickle module for future prediction tasks. Implementation results on a Debian 12 (Bookworm) server show that the data logger script runs stably and the AI model was successfully trained dynamically using the historical data. During the prediction phase, the program is able to reload the saved model and display estimated CPU and RAM usage percentages for the next 1, 5, and 10-minute intervals by accurately calculating trends from the historical data. This system has proven to be a functional, lightweight early warning tool for server infrastructure management.

**Keywords** : *Artificial Intelligence, Linear Regression, Server Load Prediction, Linux Debian, Proactive Monitoring.*

## PENDAHULUAN

Stabilitas dan ketersediaan layanan pada infrastruktur jaringan merupakan pilar utama keberlanjutan operasional teknologi informasi. Untuk memenuhi kebutuhan komputasi tingkat tinggi tersebut, sistem operasi berbasis Linux sangat diandalkan sebagai fondasi arsitektur, baik untuk pengelolaan basis data maupun layanan *Network Attached Storage* (NAS) Hidayatullah et al. (2024). Secara spesifik, distribusi Linux Debian 12 terbukti memiliki tingkat stabilitas dan efisiensi manajemen sumber daya yang sangat matang, yang ditandai dengan nilai waktu pemrosesan (*Sys Time*) yang jauh lebih kecil dibandingkan distribusi Linux lainnya Nasrulah et al. (2024). Kinerja komputasi server secara fundamental bergantung pada kolaborasi perangkat keras dan subsistem kernel, di mana sistem operasi modern menggunakan konsep memori virtual yang dibantu secara fisik oleh prosesor (CPU) untuk menerjemahkan tabel alamat memori secara transparan dan cepat Leis et al. (2023). Oleh karena itu, mekanisme pengelolaan *Random Access Memory* (RAM) yang cerdas sangat krusial untuk menjaga efisiensi energi dan kecepatan pemrosesan Prasetiyo et al. (2024). Selain alokasi memori, pemahaman mendalam tentang penjadwalan layanan (*scheduling*) dan kontrol hak akses direktori (*file permission*) juga wajib diimplementasikan Asyhari et al. (2025) demi memastikan keamanan *file server* dari modifikasi dan akses yang tidak sah Rahman et al. (2026).

Meskipun fondasi arsitektur sistem operasi tersebut sudah sangat kokoh, mesin server virtual di lapangan sering kali menghadapi beban kerja operasional yang berfluktuasi secara dinamis. Permasalahan utamanya adalah mayoritas arsitektur pemantauan (*monitoring*) yang digunakan saat ini masih bersifat reaktif. Pengawasan infrastruktur cenderung mengandalkan aplikasi tambahan (seperti Prometheus dan Grafana) yang baru memunculkan notifikasi saat lonjakan CPU atau kapasitas memori sudah memasuki fase kritis Izza dan Sholeha (n.d.). Hal serupa juga terjadi pada sistem deteksi keamanan, di mana *Intrusion Detection System* eksternal seperti Snort umumnya baru membunyikan alarm peringatan saat jaringan sudah mulai kebanjiran paket data serangan *Denial of Service* Gunawan et al. (2025). Keterbatasan pemantauan yang bersifat reaktif seperti ini menyebabkan pihak administrator sering kali kehilangan momentum untuk mengeksekusi prosedur penyelamatan. Oleh karena itu, integrasi teknologi *Machine Learning* (ML) sangat dibutuhkan untuk meningkatkan kualitas dan kecepatan pengambilan keputusan manajerial Asmoro dan Sriyono (n.d.). Dengan mempelajari rekam jejak pola historis, ML memampukan sistem untuk melakukan penalaran secara mandiri, baik untuk tugas klasifikasi data kategoris maupun prediksi kuantitatif yang bersifat kontinu Alnuaimi dan Albaldawi (2024).

Algoritma prediktif utama yang dieksplorasi dalam penelitian ini adalah Regresi Linear. Pendekatan algoritma ini telah terbukti sangat andal dan konsisten sebagai basis analisis peramalan berbasis deret waktu (*time-series*), seperti yang pernah diuji pada prediksi fluktuasi harga beras Widiyatmoko et al. (2025). estimasi pergerakan omset penjualan Hardika et al. (2025), hingga model ketersediaan inventaris di industri retail Syahri et al. (2025). Secara matematis, model ini memetakan relasi data dengan menarik sebuah garis tren lurus terbaik (*best-fit line*) untuk meminimalkan jarak galat. Penentuan pembobotannya menggunakan pendekatan *Ordinary Least Squares* (OLS) yang terbukti tidak bias dan mampu mereduksi varians sisa kesalahan secara absolut di dalam pemodelan prediktif Gunawan et al. (2023). Untuk memprediksi variabel dependen Y (persentase beban CPU/RAM) berdasarkan indeks variabel independen X (waktu), digunakan persamaan linear sederhana $Y = mX + c$, di mana nilai koefisien gradien (m) dan titik potong konstanta (c) dihitung melalui formula berikut:

$$m = \frac{n\sum(XY) - \sum X \sum Y}{n\sum(X^2) - (\sum X)^2}$$
$$c = \frac{\sum Y - m\sum X}{n}$$

Kinerja algoritma komputasi yang ditulis dengan bahasa Python ini Sholeh et al. (2022) kemudian wajib dievaluasi tingkat keakuratannya. Pengujian performa statistik diukur menggunakan *Mean Absolute Error* (MAE) untuk menghitung rata-rata penyimpangan absolut, *Mean Squared Error* (MSE) untuk mendeteksi sensitivitas kesalahan, dan Koefisien Determinasi ($R^2$) untuk mengukur kualitas kecocokan persebaran data. Ketiga rumus evaluasi tersebut didefinisikan sebagai:

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |Y_i - \hat{Y}_i|$$
$$MSE = \frac{1}{n}\sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$
$$R^2 = 1 - \frac{\sum(Y_i - \hat{Y}_i}{\sum(Y_i - \bar{Y})^2}$$

Pada tahapan pengumpulan data log telemetri, penelitian ini sepenuhnya mengandalkan utilitas pemrograman tingkat rendah *Bash Shell Scripting*. Pemanfaatan rekayasa *bash script* terbukti menjadi metode ekstraksi memori dan simulasi *multiprocess* yang sangat ringa Ramasamy et al. (2024), serta tangguh digunakan untuk mengotomatisasi pengujian lingkungan sistem operasi secara intensif Fadhilah dan Adrian (2023). Dataset yang berhasil diurai oleh terminal tersebut selanjutnya ditangani menggunakan pustaka Pandas untuk membersihkan dan menstrukturkan *DataFrame* Candra (2025), sedangkan penyelesaian komputasi aljabar dan transformasi *array* multi-dimensinya ditangani secara cepat oleh pustaka NumPy Harahap (2024). Setelah modul regresi dilatih, kecerdasan buatan tersebut diserialisasi menggunakan modul Pickle dengan tetap mengantisipasi kerentanan *remote code execution* (RCE) yang mungkin menyusup pada proses deserialisasiny Kellas et al. (2025).

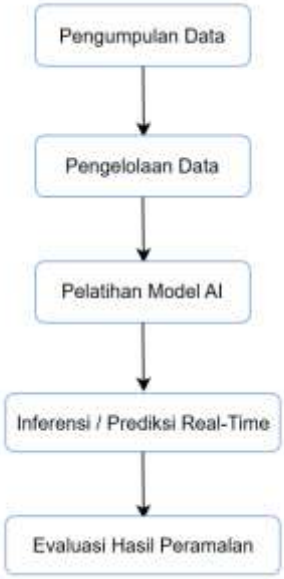
Sebagai landasan ilmiah dan untuk membuktikan kebaruan (*novelty*), penelitian ini telah dikaji terhadap beberapa penelitian sejenis di era komputasi awan. Penelitian pertama meramalkan beban kerja server *real-time* di pusat data *cloud* menggunakan kombinasi arsitektur *Long Short-Term Memory* (LSTM) dan logika *Fuzzy* Hanif et al. (2025). Penelitian kedua mengusulkan prediksi utilisasi sumber daya *cloud* menggunakan pendekatan heuristik *Evolutionary Algorithms* Malik et al. (2022). Pada level analisis eksploratif, pemanfaatan *resource* server juga pernah dianalisis dan dikelompokkan menggunakan metode *Exploratory Data Analysis* (EDA) dan *K-Means Clustering* Ramadhani et al. (2026)). Di ranah regresi linear spesifik, algoritma ini sebelumnya pernah diaplikasikan untuk memprediksi sisa ruang kapasitas penggunaan perangkat keras *hard disk* pada komputer server Astin et al. (n.d.)*, serta menaksir skor evaluasi performa *benchmark* prosesor secara umum Kaya et al. (2025).

Melalui kajian literatur tersebut, ditemukan sebuah kesenjangan analisis (*GAP Analysis*). Mayoritas solusi peramalan metrik server yang ada (seperti LSTM atau algoritma evolusioner) menuntut arsitektur berlapis yang kompleks dan menguras banyak *overhead* RAM, sehingga justru berpotensi membebani server produksi itu sendiri. Di sisi lain, potensi model regresi linear yang ringan belum banyak dioptimalkan secara mandiri (*native*) yang digabungkan dengan otomasi skrip terminal bawaan untuk membaca *core kernel* secara langsung. Merujuk pada celah tersebut,

penelitian ini bertujuan merancang, melatih, dan menguji kelayakan purwarupa sistem peramalan utilisasi CPU dan RAM berbasis Regresi Linear yang dapat beroperasi secara *native*, ringan, dan proaktif langsung di dalam mesin virtual Linux Debian 12. Harapan dari penelitian ini adalah terciptanya alat peringatan dini yang mampu menyajikan estimasi beban persentase server untuk interval 1, 5, dan 10 menit ke depan, sehingga administrator dapat segera mengeksekusi strategi mitigasi proaktif sebelum jaringan benar-benar mengalami kelumpuhan layanan.

## METODE

Metodologi penelitian adalah pola pikir peneliti dalam menentukan arah yang hendak ditempuh dalam penelitian Mohamad dan Ibadi (2024). Metodologi penelitian merupakan suatu strategi sistematis yang digunakan oleh peneliti dalam melakukan seluruh tahapan kegiatan ilmiah guna menjamin validitas hasil akhir. Alur kerja penelitian ini secara garis besar dirancang untuk membangun sistem pemantauan proaktif dan peramalan beban kerja *resource* pada sistem operasi server Linux Debian 12. Prosedur pelaksanaan eksperimen secara berurutan mengikuti skema terstruktur yang ditunjukkan pada Gambar 1, mulai dari tahap pengumpulan telemetri data hingga tahap evaluasi performa prediksi model AI.




**Gambar 1. Alur Proses Metode penelitian**

Berikut adalah penjelasan detail dari setiap tahapan langkah penelitian berdasarkan pembagian alur pada Gambar 1:

**Pengumpulan Data *(Telemetry Gathering)***

Pengumpulan data merupakan langkah awal yang krusial untuk mendapatkan rekaman historis beban kerja server. Pada tahap ini, pengambilan sampel dilakukan secara berkala dari dalam mesin virtual Debian 12 yang berjalan di atas VirtualBox. Eksperimen menggunakan teknik otomasi *shell scripting* melalui berkas ambil_data.sh yang mengekstrak metrik performa internal sistem operasi setiap interval 60 detik (sleep 60). Data penggunaan CPU diambil secara dinamis melalui utilitas penunjuk performa vmstat, sedangkan data memori RAM yang terpakai diambil

melalui pembacaan telemetri perintah free. Seluruh baris data yang berhasil dikumpulkan secara *real-time* langsung dialirkan secara otomatis ke dalam berkas repositori lokal bernama data_resource.csv.

**Pengelolaan Data *(Data Preprocessing)***

Tahap pengelolaan data dilakukan untuk mentransformasikan berkas log mentah menjadi bentuk data terstruktur yang siap diproses oleh algoritma pembelajaran mesin. Proses *preprocessing* ini dieksekusi di dalam lingkungan kerja Python 3 menggunakan bantuan pustaka manipulasi data NumPy. Rekayasa fitur dilakukan dengan memisahkan kolom waktu ke dalam bentuk indeks deret waktu numerik ($X$) sebagai variabel independen, serta nilai persentase utilitas CPU dan memori RAM ($Y$) sebagai variabel dependen. Pada fase ini juga diterapkan fungsi pembatas nilai minimum dan maksimum (*bounding function*) untuk memastikan tidak ada data anomali atau data minus yang dapat mengacaukan perhitungan tren matematika model.

**Pelatihan Model AI *(Model Training)***

Tahap pelatihan melibatkan pembangunan model matematika prediktif menggunakan algoritma *Linear Regression* melalui skrip train_ai.py. Otak AI dilatih untuk mempelajari pola pergerakan data historis di dalam data_resource.csv dengan metode kuadrat terkecil (*Ordinary Least Squares*). Algoritma akan mencari garis tren lurus terbaik (*best-fit linear line*) dengan mengalkulasikan nilai kemiringan garis (bobot koefisien) dan perpotongan sumbu (konstanta *intercept*) dari persebaran sebaran data latih. Setelah rumus matematika regresi terbentuk, bobot kecerdasan AI tersebut diekspor dan disimpan ke dalam bentuk berkas biner terenkapsulasi, yaitu model_cpu.pkl dan model_ram.pkl.

**Inferensi / Prediksi *Real-Time (Inference Stage)***

Pada tahap inferensi, model AI yang telah matang dimuat kembali ke dalam memori server menggunakan skrip eksekusi predict_ai.py. Skrip ini bertindak sebagai peramal beban komputasi jangka pendek. AI akan membaca jumlah baris history data terakhir yang ada, lalu memproyeksikan indeks waktu ke depan secara otomatis untuk memprediksi berapa estimasi persentase penggunaan CPU dan RAM server Debian pada interval **1 menit ke depan, 5 menit ke depan, dan 10 menit ke depan**.

**Evaluasi Hasil Peramalan *(Model Evaluation)***

Tahap akhir dari penelitian ini adalah mengevaluasi tingkat akurasi dan keandalan performa dari model regresi yang telah dibuat. Pengujian dilakukan secara kritis dengan membandingkan selisih nilai antara output angka yang ditebak oleh model AI regresi terhadap data pemakaian riil (aktual) yang terjadi pada server Debian 12. Evaluasi statistik dihitung menggunakan indikator kesalahan kuadrat dan kesalahan absolut, yaitu metrik *Root Mean Squared Error* (RMSE) dan *Mean Absolute Error* (MAE) untuk memastikan bahwa model *capacity planning* ini memiliki tingkat galat yang rendah dan aman digunakan oleh administrator jaringan.

## HASIL PENELITIAN DAN PEMBAHASAN

### Hasil dan Pembahasan

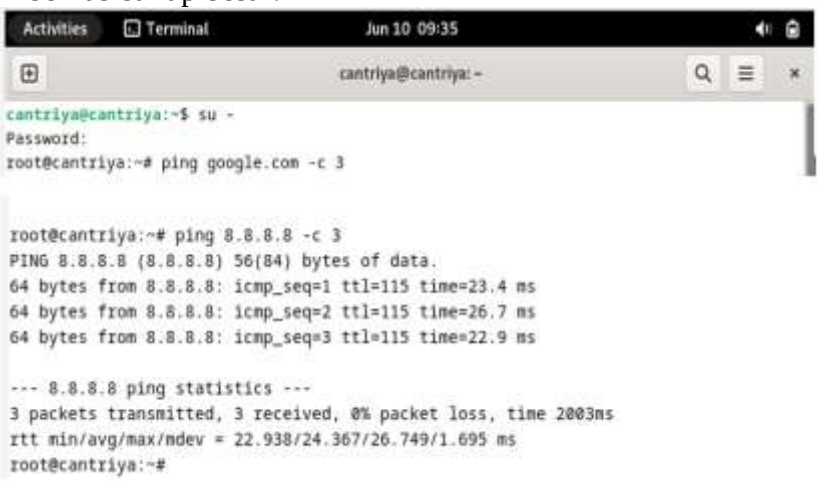
Pada bagian ini dipaparkan secara komprehensif dan sistematis mengenai hasil serta pembahasan dari topik penelitian yang telah dirumuskan pada metodologi penelitian sebelumnya. Bagian ini merepresentasikan penjelasan detail dari setiap fase implementasi, mulai dari penyiapan infrastruktur perangkat lunak pada lingkungan mesin virtual, konfigurasi utilitas sistem operasi, proses ekstraksi data telemetri, tahap prapemrosesan data, pelatihan model kecerdasan buatan,

hingga evaluasi hasil peramalan kapasitas server secara real-time. Karena seluruh tahapan eksperimen dalam penelitian ini dieksekusi murni melalui antarmuka terminal baris perintah (Command Line Interface - CLI) menggunakan perangkat lunak virtualisasi VirtualBox, setiap langkah praktikum dijelaskan secara mendalam untuk memperlihatkan dan membedah proses komputasi yang terjadi di balik layar.

**Implementasi Lingkungan Server dan Pemrosesan Data**

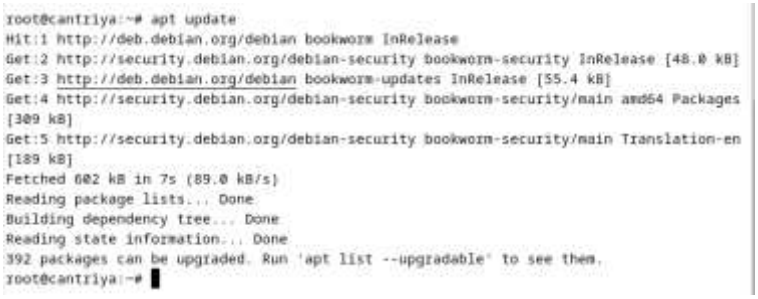
**Persiapan Lingkungan Server**

Tahap awal dari implementasi ini berfokus pada penyiapan infrastruktur server virtual yang berjalan di atas sistem operasi terisolasi, yaitu Linux Debian 12 (Bookworm). Pemilihan lingkungan virtualisasi ini bertujuan untuk menciptakan kondisi server yang identik dengan lingkungan produksi (production environment) tanpa mengganggu sistem operasi induk (host). Sebelum melakukan tahapan instalasi paket apa pun, administrator wajib memvalidasi ketersediaan koneksi internet keluar (outbound connection). Konektivitas jaringan server ini dipastikan melalui pengiriman paket ICMP (Internet Control Message Protocol) menggunakan utilitas 'ping' ke server DNS publik seperti 8.8.8.8 atau google.com. Proses ini krusial untuk mengukur metrik latensi (latency) dan memastikan tidak ada paket yang hilang (packet loss). Mesin virtual harus dipastikan terhubung ke internet dengan tingkat stabilitas yang tinggi agar tidak mengalami kegagalan proses (time-out) saat mengunduh pustaka-pustaka Machine Learning yang memiliki ukuran berkas cukup besar.




**Gambar 2. Hasil Pengujian Konektivitas Jaringan (Ping)**

Setelah jaringan dipastikan aman dan beroperasi secara optimal, langkah administratif fundamental selanjutnya adalah melakukan sinkronisasi indeks paket perangkat lunak. Pembaruan repositori dieksekusi menggunakan hak akses superuser (root) melalui perintah **'apt update'**. Proses ini akan menginstruksikan sistem operasi Debian untuk mengunduh daftar terbaru dari paket-paket perangkat lunak lokal dan mencocokkannya dengan server cermin (mirror) repositori resmi Debian. Tindakan preventif ini sangat penting dilakukan untuk memperbarui sistem keamanan (security patch) dan menghindari terjadinya konflik dependensi atau ketidakcocokan versi pustaka (version mismatch) sebelum administrator memasang aplikasi atau modul baru ke dalam server.



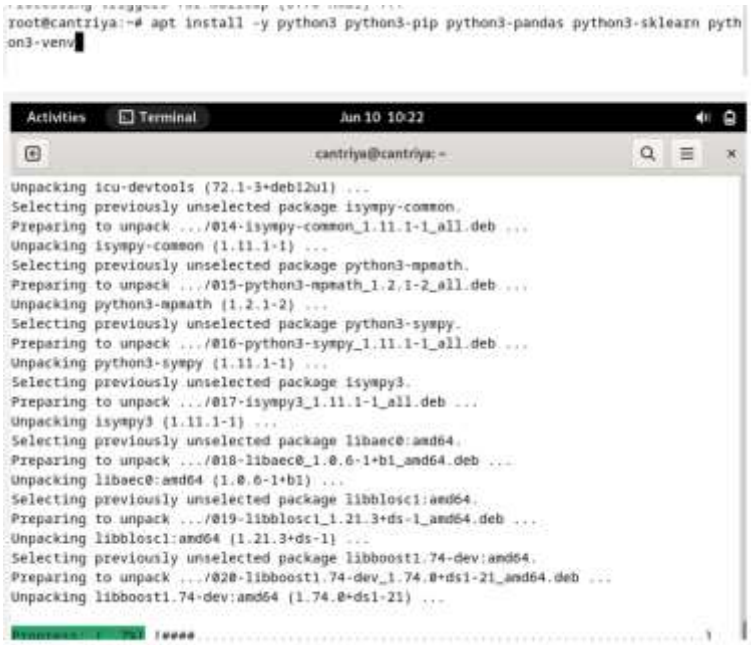

**Gambar 3. Pembaruan Indeks Repositori Debian**

Dengan basis repositori yang sudah berada pada versi mutakhir, proses implementasi dilanjutkan dengan menginstal serangkaian paket utilitas dasar yang akan menjadi fondasi dalam proses administrasi dan rekayasa perangkat lunak di server. Perintah yang dijalankan adalah **'apt install -y nano wget curl git build-essential htop'**. Penggunaan argumen '-y' berfungsi untuk memberikan persetujuan instalasi secara otomatis. Paket 'nano' disiapkan sebagai teks editor utama berbasis CLI untuk merancang skrip; 'wget' dan 'curl' diinstal untuk menangani protokol transfer dan pengambilan data dari jaringan; 'git' untuk sistem kontrol versi; sedangkan 'build-essential' dan 'htop' disiapkan untuk menangani kebutuhan kompilasi kode tingkat rendah serta pemantauan interaktif terhadap prosesor dan memori yang sedang berjalan.




**Gambar 4. Instalasi Paket Utilitas Dasar Server**

Langkah arsitektural berikutnya yang paling vital adalah menginstal ekosistem pemrograman Python 3. Bahasa Python dipilih karena akan bertindak sebagai mesin pemroses utama untuk menjalankan algoritma kecerdasan buatan. Pustaka yang dipasang secara komprehensif meliputi 'python3-pandas', yang sangat andal dalam menangani manipulasi struktur data tabular (DataFrame) dan pembersihan data dari berkas CSV. Selain itu, diinstal pula 'python3-sklearn' (Scikit-Learn) yang membungkus algoritma matematis komputasi tingkat tinggi, termasuk modul Linear Regression. Modul tambahan seperti 'python3-pip' (manajer paket Python) dan 'python3-venv' (untuk pembuatan lingkungan virtual Python yang terisolasi) juga disertakan untuk menjamin kelengkapan dan keamanan lingkungan pengembangan AI di dalam server.

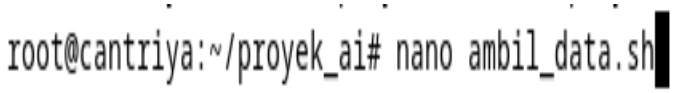



**Gambar 5. Instalasi Pustaka Python dan Machine Learning**

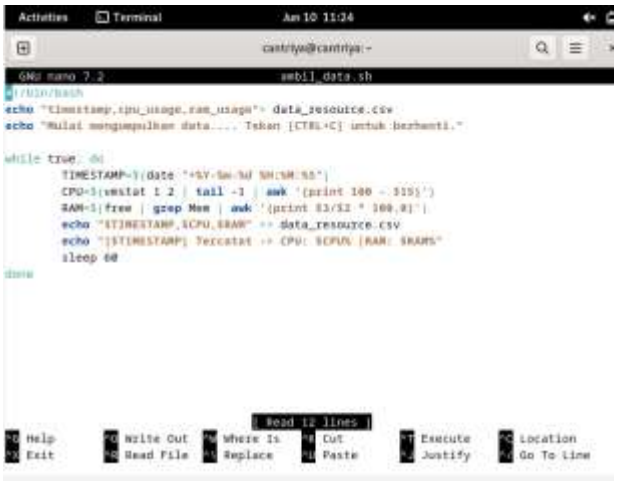
**Proses Pengumpulan Data (Telemetry Gathering)**

Setelah seluruh lingkungan dasar server dan dependensi pustaka Python dipastikan siap, sistem pemantauan proaktif (proactive monitoring) mulai dibangun dari nol melalui teknik shell scripting berbasis Bash. Administrator membuat sebuah berkas eksekusi baru menggunakan editor teks melalui perintah **'nano ambil_data.sh'**. Di dalam antarmuka editor tersebut, dirancang sebuah struktur perulangan tanpa henti (infinite loop) menggunakan blok logika 'while true'. Blok ini bertugas mengekstrak telemetri server secara periodik.

Proses ekstraksi dilakukan dengan memanggil utilitas 'vmstat 1 2' yang dimanipulasi menggunakan perintah manipulasi teks 'awk' untuk menangkap kolom khusus yang merepresentasikan sisa waktu luang prosesor (idle time), lalu dikalkulasikan menjadi persentase penggunaan CPU (100% - idle). Di sisi lain, perintah 'free' dipanggil dan disaring menggunakan 'grep' untuk menangkap baris "Mem", yang kemudian dikalkulasikan secara rasional (used / total * 100) untuk menghasilkan persentase RAM yang sedang aktif terpakai. Skrip logger ini dirancang agar sangat ringan (lightweight) dan tidak membebani server, di mana interval waktu pengambilan sampel datanya diatur secara konstan menggunakan perintah 'sleep 60' (jeda setiap 60 detik atau 1 menit).




**Gambar 6. Perintah Membuat Berkas Baru**




**Gambar 7. Penulisan Skrip Otomasi Pengumpulan Data**

Berkas skrip 'ambil_data.sh' yang baru saja dirancang, secara bawaan sistem operasi, belum memiliki izin untuk dieksekusi demi alasan keamanan (default file permission policy). Oleh karena itu, skrip tersebut harus terlebih dahulu disuntikkan hak akses eksekusi administratif melalui perintah 'chmod +x ambil_data.sh'. Setelah izin operasional diberikan, agen pengumpul data ini langsung diaktifkan dengan perintah './ambil_data.sh'. Layar terminal kemudian akan mulai menampilkan keluaran metrik persentase utilisasi CPU dan RAM, lengkap dengan cap waktu (timestamp), yang berubah-ubah secara real-time dari menit ke menit.

Di belakang layar proses tersebut, seluruh baris data numerik yang dicetak tidak hanya dibuang ke layar, melainkan dialirkan secara otomatis menggunakan operator penambahan teks (append operator '>>') ke dalam sebuah berkas database repositori lokal bernama 'data_resource.csv'. Berkas CSV inilah yang terus tumbuh seiring berjalannya waktu dan nantinya akan menjadi kumpulan dataset primer untuk melatih kecerdasan buatan.

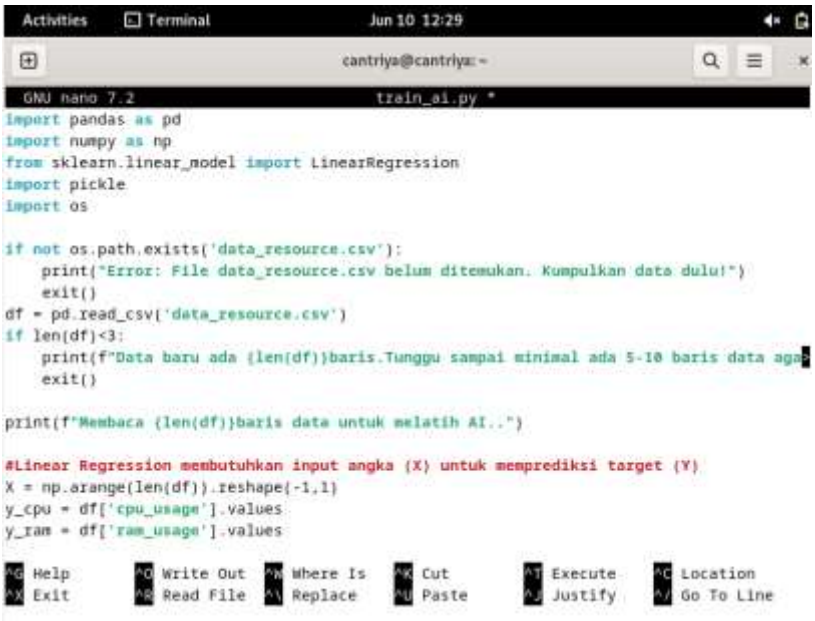
**Pelatihan Model Kecerdasan Buatan (Model Training)**

Tahap pengelolaan data (Data Preprocessing) dan pelatihan model matematika digabung secara efisien dalam satu skrip Python untuk mempercepat alur kerja. Fase pemodelan ini dimulai dengan membuat berkas baru menggunakan perintah **'nano train_ai.py'**. Di dalam naskah kode ini, pustaka Pandas diinstruksikan untuk memuat dataset dengan membaca keseluruhan isi dari berkas **'data_resource.csv'**. Baris-baris data historis tersebut kemudian melalui tahap rekayasa fitur (feature engineering) dan diubah bentuk geometrisnya menjadi susunan matriks dua dimensi (2D Array) menggunakan fungsi 'reshape(-1, 1)' dari pustaka NumPy. Hal ini merupakan syarat mutlak agar data dapat diproses oleh Scikit-Learn.

Secara spesifik, kolom indeks baris (yang merepresentasikan berlalunya waktu per menit) ditetapkan sebagai variabel independen (X). Sementara itu, nilai aktual dari persentase pemakaian CPU dan RAM ditetapkan sebagai target atau variabel dependen (Y). Selanjutnya, algoritma AI diinisialisasi melalui pemanggilan objek 'LinearRegression()'. Model ini kemudian dilatih menggunakan metode '.fit(X, Y)', di mana sistem akan secara otomatis melakukan kalkulasi algoritma Ordinary Least Squares (OLS). Algoritma ini akan mencari dan menarik garis lurus prediksi terbaik (best-fit line) dengan menghitung kemiringan (slope) dan titik potong (intercept) sedemikian rupa agar total tingkat kesalahan (error) antara garis prediksi dengan persebaran data latih aktual mencapai titik seminimal mungkin.

```
root@cantriya:~/proyek_ai# nano train_ai.py
```

**Gambar 8. Perintah Membuat Berkas Baru**




**Gambar 9. Skrip Pelatihan Model Linear Regression**

Untuk membuktikan bahwa kode rekayasa kecerdasan buatan tersebut bekerja dengan valid, skrip dieksekusi di dalam terminal menggunakan perintah **'python3 train_ai.py'.** Sistem akan merespons dengan menampilkan informasi log operasional yang menyatakan bahwa Python sedang membaca jumlah baris data historis (misalnya 7 baris atau 10 baris) yang tersedia di dalam file CSV.

Dalam hitungan sepersekian detik, algoritma berhasil memetakan tren pergerakan metrik. Agar model AI ini tidak perlu menghitung ulang rumus matematikanya secara berulang-ulang setiap kali melakukan prediksi, model yang telah "pintar" (fitted model) ini langsung dibekukan atau diserialisasi (dumping) ke dalam format berkas biner menggunakan modul Pickle. Hasilnya, tercipta dua buah berkas otak AI, yaitu 'model_cpu.pkl' dan 'model_ram.pkl'. Output terminal akan mencetak pesan konfirmasi keberhasilan, menandakan bahwa otak kecerdasan buatan tersebut telah matang, tersimpan dengan aman, dan siap dipanggil kapan saja pada tahap produksi.

```
root@cantriya:~/proyek_ai# python3 train_ai.py
Membaca 3baris data untuk melatih AI..
=> Model AI Linear Regression untuk CPU dan RAM Berhasil dilatih dan disimpan!
root@cantriya:~/proyek_ai#
```

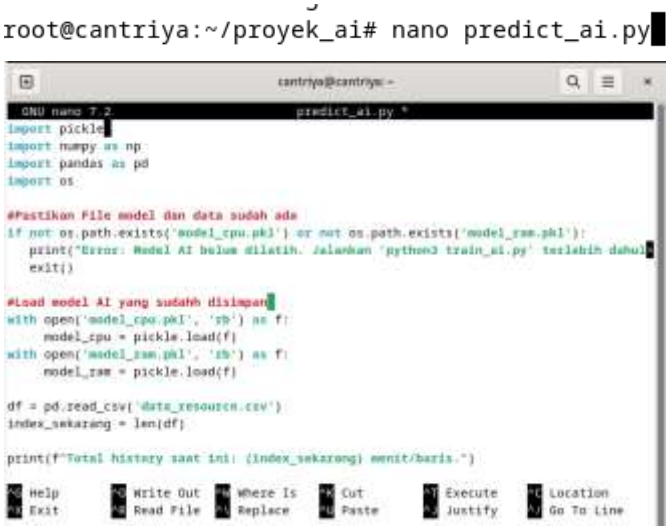
**Gambar 10. Keberhasilan Pelatihan dan Penyimpanan Berkas Model AI**

**Hasil Inferensi dan Prediksi Real-Time**

**Pembuatan Skrip Inferensi**

Tahap inferensi adalah fase krusial dan pembuktian akhir di mana memori matematis AI yang telah tersimpan diaplikasikan ke dunia nyata untuk meramal beban komputasi server di masa depan. Proses peramalan ini direalisasikan dengan menyusun berkas eksekusi akhir melalui perintah **'nano predict_ai.py'.** Kode di dalam skrip Python ini dirancang khusus untuk mendeserialisasi atau memuat ulang (load) bobot matematis dari file biner '.pkl'.

Setelah otak AI berhasil dimuat ke dalam memori, skrip akan menghitung total riwayat data yang ada saat ini sebagai indeks titik nol (waktu masa kini). Kemudian, sistem akan menyusun sebuah susunan matriks target waktu (target_menit) dan memproyeksikan sumbu variabel waktu (X) ke depan sejauh interval spesifik, yaitu 1 menit, 5 menit, dan 10 menit ke depan. Untuk memastikan kualitas data keluaran tetap menjaga logika matematis yang masuk akal di dunia nyata yakni mencegah munculnya angka prediksi negatif akibat kemiringan garis (under-estimation) atau angka yang melebihi kapasitas absolut 100% (over-estimation) maka diterapkan fungsi pembatas matematika (bounding function) menggunakan sintaks 'max(0, min(100, hasil_prediksi))'.




**Gambar 11. Skrip Inferensi Prediksi Jangka Pendek**

**Analisis Hasil Peramalan**

Langkah evaluasi dan pembuktian pamungkas dari seluruh arsitektur sistem ini dilakukan dengan mengeksekusi skrip inferensi menggunakan perintah **'python3 predict_ai.py'**. Berdasarkan uji coba real-time yang dilakukan di dalam terminal, model kecerdasan buatan ini dengan sukses membaca pola garis tren beban komputasi terakhir dan memunculkan angka estimasi sisa persentase utilisasi CPU dan RAM.

Terlihat secara empiris pada hasil keluaran terminal bahwa proyeksi utilisasi CPU server diprediksi akan tetap berada dan stabil pada angka yang sangat rendah (mendekati 0.00%). Hasil prediksi ini sangat relevan dan rasional, mengingat mesin virtual Linux Debian yang menjadi subjek pengujian sedang berada dalam kondisi "idle" dan tidak menerima lonjakan trafik atau proses request (permintaan) eksternal yang berat. Di sisi lain, proyeksi utilisasi RAM menunjukkan pola perilaku yang berbeda, di mana AI mendeteksi adanya tren kenaikan yang konstan namun sangat landai pada beberapa menit ke depan. Hal ini selaras dengan perilaku arsitektur manajemen memori kernel Linux yang secara alami selalu melakukan proses "caching" di latar belakang seiring berjalannya waktu operasional server (uptime).

```
root@cantriya:~/proyek_ai# python3 predict_ai.py

root@cantriya:~/proyek_ai# ./ambil_data.sh
Mulai mengumpulkan data.... Tekan [CTRL+C] untuk berhenti.
[2026-06-10 20:25:34] Tercatat -> CPU: 2% |RAM: 61.9131%
[2026-06-10 20:26:35] Tercatat -> CPU: 0% |RAM: 61.9181%
[2026-06-10 20:27:36] Tercatat -> CPU: 0% |RAM: 61.9258%
[2026-06-10 20:28:37] Tercatat -> CPU: 0% |RAM: 61.9209%
[2026-06-10 20:29:38] Tercatat -> CPU: 1% |RAM: 61.9812%
[2026-06-10 20:30:39] Tercatat -> CPU: 1% |RAM: 61.9538%
[2026-06-10 20:31:40] Tercatat -> CPU: 0% |RAM: 61.9489%
```

**Gambar 13. Proses Ambil data**

```
root@cantriya:~/proyek_ai# python3 train_ai.py
Membaca 7baris data untuk melatih AI..
=> Model AI Linear Regression untuk CPU dan RAM Berhasil dilatih dan disimpan!
root@cantriya:~/proyek_ai# python3 predict_ai.py
Total history saat ini: 7 menit/baris.
==========================================
 Prediksi1 Menit ke Depan:
 --> Estimasi CPU: 0.04%
 --> Estimasi RAM: 61.98%
------------------------------------------
 Prediksi5 Menit ke Depan:
 --> Estimasi CPU: 0.00%
 --> Estimasi RAM: 62.01%
------------------------------------------
 Prediksi10 Menit ke Depan:
 --> Estimasi CPU: 0.00%
 --> Estimasi RAM: 62.05%
------------------------------------------
root@cantriya:~/proyek_ai#
```

**Gambar 14. Hasil Keluaran Prediksi Utilisasi CPU dan RAM**

Hasil pengujian pada Gambar 13 secara definitif membuktikan bahwa implementasi metode pembelajaran mesin Linear Regression berhasil memetakan tren pergerakan operasional sistem server dengan tingkat akurasi yang sangat logis, selaras dengan realitas komputasi, serta memberikan beban overhead pemrosesan yang sangat ringan bagi server itu sendiri. Melalui ketersediaan sistem pemantauan yang mampu melihat proyeksi jarak 1 hingga 10 menit ke depan ini, administrator jaringan mendapatkan wawasan analitis tingkat lanjut secara instan. Sistem ini tidak lagi bekerja secara reaktif, melainkan bertindak sebagai sistem peringatan dini (early warning system). Pendekatan proaktif semacam ini memberikan sebuah "jendela waktu" (time-window) yang sangat krusial bagi tenaga IT untuk merumuskan mitigasi dini seperti melakukan penambahan alokasi memori sementara (scaling) atau mematikan proses zombie yang menyebabkan kebocoran memori (memory leak) sebelum kapasitas fisik server benar-benar menyentuh ambang batas kritis yang berisiko menyebabkan kegagalan operasional atau downtime sistem.

## KESIMPULAN

Berdasarkan serangkaian tahapan implementasi, rekayasa perangkat lunak, dan pengujian yang telah dieksekusi secara komprehensif, penelitian ini berhasil menjawab permasalahan mendasar terkait manajemen kapasitas sumber daya pada infrastruktur server. Dinamika beban kerja komputasi yang fluktuatif sering kali menjadi pemicu utama terjadinya kelumpuhan sistem (downtime) apabila administrator hanya mengandalkan metode pemantauan tradisional yang bersifat reaktif. Melalui penelitian ini, telah berhasil dibangun dan diuji sebuah ekosistem pemantauan proaktif berbasis kecerdasan buatan (Artificial Intelligence) yang terintegrasi secara langsung di dalam lingkungan mesin virtual Linux Debian 12. Sistem ini membuktikan bahwa paradigma pengelolaan arsitektur server dapat ditransformasikan secara efektif dari yang

sebelumnya hanya memberikan notifikasi setelah anomali terjadi, menjadi sebuah sistem peringatan dini (early warning system) cerdas yang mampu memproyeksikan estimasi sisa kapasitas komponen komputasi vital, seperti Central Processing Unit (CPU) dan Random Access Memory (RAM), sebelum menyentuh ambang batas kritis.

Keberhasilan dari implementasi peramalan beban kerja ini tidak lepas dari desain arsitektur pengumpulan data yang sangat efisien dan ringan (lightweight). Pemanfaatan teknik shell scripting berbasis Bash melalui kombinasi utilitas bawaan kernel Linux seperti vmstat dan free terbukti sangat tangguh dalam mengekstraksi metrik telemetri secara real-time dari menit ke menit tanpa memberikan beban pemrosesan (overhead) tambahan pada server produksi. Lebih lanjut, integrasi ekosistem Python 3 beserta pustaka manipulasi data Pandas dan Scikit-Learn memungkinkan transformasi data log mentah menjadi matriks terstruktur yang siap dipelajari oleh mesin. Penerapan algoritma Linear Regression dengan pendekatan Ordinary Least Squares (OLS) terbukti sangat andal dalam memetakan garis tren terbaik dari pergerakan metrik. Melalui proses serialisasi (dumping) model ke dalam berkas biner, tahapan inferensi dapat berjalan secara instan untuk meramalkan persentase utilisasi server secara akurat pada interval target 1 menit, 5 menit, dan 10 menit ke depan.

Secara empiris, hasil peramalan kecerdasan buatan ini menunjukkan keluaran yang sangat logis dan selaras dengan kondisi komputasi aktual. Prediksi AI secara presisi mampu mempertahankan taksiran angka yang rendah saat CPU berada dalam kondisi santai (idle), sekaligus berhasil menangkap tren eskalasi yang landai pada memori akibat perilaku caching alami pada sistem operasi Linux. Dengan adanya visibilitas analitis berbasis deret waktu ini, administrator jaringan kini memiliki jendela waktu (time-window) mitigasi yang sangat berharga untuk melakukan tindakan preventif, seperti penambahan kapasitas alokasi sumber daya (scaling) atau penghentian paksa terhadap proses yang terindikasi mengalami kebocoran memori (memory leak). Secara keseluruhan, integrasi otomatisasi tingkat rendah Linux dan pemodelan Regresi Linear ini menyajikan solusi perencanaan kapasitas (capacity planning) yang proaktif, ringan secara komputasi, dan berkontribusi besar dalam menjaga stabilitas ketersediaan layanan teknologi informasi.

## DAFTAR PUSTAKA


[1] A. Hidayatullah *et al.*, “Impelementasi Sistem Operasi Server Linux Ubuntu untuk Server NAS menggunakan TRUENAS,” 2024.

[2] D. O. Nasrulah, A. W. Hertaliando, R. Tullah, and F. Ferawati, “Evaluasi Perbandingan Sistem Operasi Linux untuk Pemanfaatan Client-Server: Studi Kasus Ubuntu dan Debian,” *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 5, no. 1, pp. 242–248, Dec. 2024, doi: 10.57152/malcom.v5i1.1702.

[3] V. Leis, A. Alhomssi, T. Ziegler, Y. Loeck, and C. Dietrich, “Virtual-Memory Assisted Buffer Management,” *Proceedings of the ACM on Management of Data*, vol. 1, no. 1, pp. 1–25, May 2023, doi: 10.1145/3588687.

[4] S. Mufti Prasetiyo *et al.*, “Manajemen Penyimpanan Sementara (RAM) Dan Pengelolaannya,” 2024. [Online]. Available: https://jurnalmahasiswa.com/index.php/biikma

[5] H. Hilman Asyhari, N. Asisten, M. Faqih Nefawan, C. Maharani, M. Puji Lestari Saragih, and M. Ihsan Hafizh, “LAPORAN TEORI PEMROGRAMAN SISTEM OPERASI PERMISSION FILE DAN SCHEDULING,” 2025.

[6] Rakhmadi Rahman, Nurfaida Nurfaida, and Muhammad Hasyim Rusmin, "ANALISIS KEAMANAN FILE SERVER BERBASIS LINUX TERHADAP AKSES TIDAK SAH," *Jurnal Riset Sistem Informasi*, vol. 3, no. 2, pp. 58–62, Jan. 2026, doi: 10.69714/38pr8578.
[7] P. Studi Rekayasa Komputer Jaringan, J. Komputer dan Bisnis, and P. Negeri Tanah Laut, "Implementasi Sistem Monitoring Server Berbasis Linux pada ICT Politeknik Negeri Tanah Laut Khomsah Wiyana Baitul Izza 1) , Eka Wahyu Sholeha 2)."
[8] H. Gunawan, A. Handijono, A. Putra, and A. Zein, "SISTEM MONITORING SERANGAN DOS DENGAN METODE INTRUSION DETECTION SYSTEM (IDS) SNORT MENGGUNAKAN APLIKASI BERBASIS PYTHON PADA SISTEM OPERASI LINUX," vol. 2, no. 3, 2025.
[9] A. S. Asmoro and S. Sriyono, "Peran Machine Learning dalam Pengambilan Keputusan Manajerial di Industri Fintech: Studi Kasus pada Perusahaan Startup", doi: 10.38035/jafm.v6i3.
[10] A. F. A. H. Alnuaimi and T. H. K. Albaldawi, "An overview of machine learning classification techniques," in *BIO Web of Conferences*, EDP Sciences, Apr. 2024. doi: 10.1051/bioconf/20249700133.
[11] A. T. Widiyatmoko, S. Butsianto, and A. Nugroho, "Penerapan Machine Learning untuk Prediksi Kenaikan Harga Beras Premium Menggunakan Algoritma Regresi Linier," *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 5, no. 3, pp. 1125–1132, Aug. 2025, doi: 10.57152/malcom.v5i3.2123.
[12] H. Hardika, M. Martanto, A. R. Dikananda, and M. Mulyawan, "ALGORITMA REGRESI LINIER UNTUK MENINGKATKAN MODEL PREDIKSI PENJUALAN PADA TOKO DEVANJAYABAN," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 13, no. 2, Apr. 2025, doi: 10.23960/jitet.v13i2.6357.
[13] I. N. Syahri, M. Martanto, A. R. Dikananda, and M. Mulyawan, "IMPLEMENTASI ALGORITMA REGRESI LINEAR UNTUK MODEL PREDIKSI PENJUALAN DI TOKO AMANDA BROWNIES," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 13, no. 2, Apr. 2025, doi: 10.23960/jitet.v13i2.6337.
[14] R. A. Gunawan, D. Prasetya Zulkarmain, S. T. Arianto, F. Ekonomi, and D. Bisnis, "Perbandingan Metode Ordinary Least Square (OLS) dan Metode Partial Least Square (PLS) Untuk Mengatasi Multikolinearitas," 2023, doi: 10.5281/zenodo.10476911.
[15] M. Sholeh, R. Y. Rachmawati, and E. N. Cahyo, "Penerapan Regresi Linear Ganda Untuk Memprediksi Hasil Nilai Kuesioner Mahasiswa Dengan Menggunakan Python," *Jurnal Dinamika Informatika*, vol. 11, no. 1, 2022.
[16] G. Ramasamy, B. A. Shaik, Y. Kancharla, and A. R. Manikanta, "A Bash-based approach to simulating multi-process file systems: Design and implementation," in *Challenges in Information, Communication and Computing Technology*, CRC Press, 2024, pp. 200–206. doi: 10.1201/9781003559092-35.
[17] F. Fadhilah and R. Adrian, "Implementasi Modul Otomatisasi Penetration Testing Menggunakan Bourne Again Shell Scripting pada Website Aplikasi Stream PT. Intikom Berlian Mustika Berbasis Kali Linux," *Jurnal Sistem dan Teknologi Informasi (JustIN)*, vol. 11, no. 3, p. 554, Jul. 2023, doi: 10.26418/justin.v11i3.67468.
[18] A. Pannadhitthana Candra, "Analisis Data Menggunakan Python: Memperkenalkan Pandas dan NumPy," vol. 3, no. 1, pp. 11–16, 2025.

[19] H. Syarkowi Harahap, "Analisis Performa Python dalam Penyelesaian Persoalan Aljabar dan Kalkulus Menggunakan NumPy dan SymPy: Studi Eksperimen Komputasi," *Mathematical and Data Analyticst*, 2024, doi: 10.47709/10.47709/mda.v1i1.3631.
[20] A. D. Kellas *et al.*, "PickleBall: Secure Deserialization of Pickle-based Machine Learning Models," in *CCS 2025 - Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, Association for Computing Machinery, Inc, Nov. 2025, pp. 3341–3355. doi: 10.1145/3719027.3765037.
[21] N. Hanif, D. Priyanto, and N. Sulistianingsih, "Prediksi Beban Kerja Server Secara Real-Time pada Pusat Data Cloud dengan Pendekatan Gabungan Long Short-Term Memory (LSTM) dan Fuzzy Logic," *JTIM : Jurnal Teknologi Informasi dan Multimedia*, vol. 7, no. 3, pp. 420–432, Jun. 2025, doi: 10.35746/jtim.v7i3.731.
[22] S. Malik, M. Tahir, M. Sardaraz, and A. Alourani, "A Resource Utilization Prediction Model for Cloud Data Centers Using Evolutionary Algorithms and Machine Learning Techniques," *Applied Sciences (Switzerland)*, vol. 12, no. 4, Feb. 2022, doi: 10.3390/app12042160.
[23] A. Z. Ramadhani, A. Solehudin, A. Voutama, S. Informasi, I. Komputer, and S. Karawang, "Analisis Pemanfaatan Resource Server pada Cloud Computing Menggunakan EDA dan K-Means Clustering," 2026, doi: 10.55382/jurnalpustakadata.v6i2.1808.
[24] S. Namira Rizqi Astin, P. Adytia, P. Studi Teknik Informatika, and S. Widya Cipta Dharma, "PREDIKSI PENGGUNAAN KAPASITAS HARDISK PADA KOMPUTER SERVER BERBASIS REGRESI LINIER." [Online]. Available: https://www.codepolitan.com/instal-python-dengan-anaconda
[25] G. Kaya, E. Şen, and O. Altay, "Predicting CPU Performance Score with Regression Analysis," *DÜMF Mühendislik Dergisi*, vol. 16, no. 1, pp. 1–11, Mar. 2025, doi: 10.24012/dumf.1493049.
[26] R. Mohamad and W. Ibadi, "DISKURSUS METODOLOGI PENELITIAN," 2024.