

# PREDIKSI JUMLAH PENYEWAAN SEPEDA MENGGUNAKAN MACHINE LEARNING DAN DEEP LEARNING

<b>Nama Mahasiswa</b>	:	Zahy Nadhir Nashrullah
<b>Nim</b>	:	233307060
<b>Program Studi</b>	:	Teknologi Informasi
<b>Mata Kuliah</b>	:	Data Science
<b>Dosen Pengampu</b>	:	Gus Nanang Syaifuddiin, S.Kom., M.Kom.
<b>Tahun Akademik</b>	:	2025 / 5
<b>Link Video Pembahasan</b>	:	<a href="https://youtu.be/_mTu-6IgZU">https://youtu.be/_mTu-6IgZU</a>
<b>Link GITHUB Repository</b>	:	<a href="https://github.com/zahynadhirnashrullah/UAS-Bike-Sharing-Prediction.git">https://github.com/zahynadhirnashrullah/UAS-Bike-Sharing-Prediction.git</a>

## 1. LEARNING OUTCOMES

Pada proyek ini, mahasiswa diharapkan dapat:

1. Memahami konteks masalah bisnis penyewaan sepeda dan merumuskan problem statement secara jelas.
2. Melakukan analisis dan eksplorasi data (EDA) untuk melihat pola penyewaan berdasarkan waktu dan cuaca.
3. Melakukan data preparation (cleaning, scaling, splitting) yang sesuai dengan dataset Bike Sharing.
4. Mengembangkan tiga model machine learning yang terdiri dari (WAJIB):
  - a. Model Baseline: Linear Regression
  - b. Model Advanced: Random Forest Regressor
  - c. Model Deep Learning: Multilayer Perceptron (MLP)
5. Menggunakan metrik evaluasi yang relevan untuk regresi (MAE, RMSE).
6. Melaporkan hasil eksperimen secara ilmiah dan sistematis.
7. Mengunggah seluruh kode proyek ke GitHub.
8. Menerapkan prinsip software engineering dalam pengembangan proyek.

## **2. PROJECT OVERVIEW**

### **2.1 Latar Belakang**

Sistem berbagi sepeda (bike sharing system) dapat dipandang sebagai bentuk modern dari layanan penyewaan sepeda konvensional. Seluruh proses berlangsung otomatis, mulai pendaftaran pengguna, peminjaman sepeda, dan pengembalian. Kehadiran sistem memberikan manfaat nyata bagi kota, seperti membantu mengurangi kemacetan, menurunkan tingkat polusi udara, serta mendorong gaya hidup yang lebih sehat di kalangan masyarakat perkotaan.

Meski demikian, pengelola sistem berbagi sepeda dihadapkan tantangan besar, yaitu menjaga ketersediaan sepeda tetap seimbang di setiap stasiun. Jumlah permintaan sewa dapat berubah-ubah secara signifikan dan dipengaruhi oleh berbagai faktor eksternal, seperti cuaca, musim, hari libur, serta waktu dalam sehari. Ketika permintaan tidak dapat diprediksi dengan baik, kondisi dapat memicu kekurangan sepeda di satu lokasi dan penumpukan di lokasi lainnya.

Untuk mengatasi permasalahan tersebut, pemanfaatan machine learning menjadi salah satu pendekatan yang efektif dalam memprediksi jumlah penyewaan sepeda di masa mendatang. Penelitian dilakukan oleh Fanaee-T dan Gama (2014) menunjukkan data historis penyewaan yang dikombinasikan dengan informasi cuaca dapat berfungsi sebagai “sensor virtual” untuk memahami pola mobilitas perkotaan dan memprediksi kejadian tertentu. Temuan diperkuat oleh studi Gao dan Chen (2022), yang menyatakan bahwa algoritma machine learning seperti Random Forest mampu menghasilkan prediksi permintaan sepeda dengan tingkat akurasi yang lebih tinggi dibandingkan metode statistik konvensional.

Berdasarkan uraian tersebut, proyek ini bertujuan untuk mengembangkan model prediksi berbasis regresi menggunakan dataset Bike Sharing dari UCI Machine Learning Repository, mendukung pengelolaan stok sepeda yang lebih efisien dan tepat sasaran.

## **Referensi:**

1. Fanaee-T, H., & Gama, J. (2014). Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, 2(2), 113-127.
2. Gao, C., & Chen, Y. (2022). Using machine learning methods to predict demand for bike sharing. *Information and Communication Technologies in Tourism* 2022, 282-296. Springer.

### **3. BUSINESS UNDERSTANDING / PROBLEM UNDERSTANDING**

#### **3.1 Problem Statements**

Masalah yang ingin diselesaikan dalam proyek ini adalah:

1. Bagaimana memprediksi jumlah total penyewaan sepeda (casual + registered) secara akurat berdasarkan fitur cuaca (suhu, kelembaban, kecepatan angin) dan waktu (jam, hari, musim)?
2. Model machine learning mana yang memberikan performa terbaik dengan tingkat error terendah?
3. Apakah pendekatan Deep Learning mampu memberikan hasil yang lebih baik dibandingkan model konvensional pada data ini?

#### **3.2 Goals**

Tujuan dari proyek ini adalah:

1. Membangun model regresi yang mampu memprediksi variabel target cnt (jumlah total sewa) dengan akurasi yang dapat diandalkan.
2. Mengukur dan membandingkan performa tiga model: Linear Regression (Baseline), Random Forest (Advanced), dan Multilayer Perceptron (Deep Learning).
3. Menghasilkan sistem prediksi yang dapat membantu pengelola dalam pengambilan keputusan distribusi sepeda.

#### **3.3 Solution Approach**

##### **3.3.1 Model 1 – Baseline Model**

1. **Model:** Linear Regression
2. **Alasan:** model paling sederhana untuk kasus regresi. Linear Regression bekerja dengan mencari garis lurus yang paling pas mewakili hubungan antara input (cuaca)

dan output (jumlah sewa). Model cepat dilatih dan mudah dijelaskan, sehingga sangat cocok dijadikan standar pembanding awal.

### 3.3.2 Model 2 – Advanced / ML Model

1. **Model:** Random Forest Regressor
2. **Alasan:** Random Forest adalah kumpulan dari banyak Decision Tree. Model dipilih karena lebih tangguh (robust) terhadap data yang bervariasi dan mampu menangkap hubungan non-linear yang tidak bisa ditangkap oleh Linear Regression. Random Forest biasanya memberikan akurasi yang sangat tinggi untuk data berbentuk tabel.

### 3.3.3 Model 3 – Deep Learning Model

1. **Model:** Multilayer Perceptron (MLP) / Neural Network.
2. **Jenis:** Tabular Data.

**Alasan Pemilihan:** karena data berbentuk tabel angka (baris dan kolom Excel), bukan berupa gambar (yang butuh CNN) atau teks kalimat (yang butuh LSTM). MLP bekerja meniru cara kerja otak sederhana:

1. **Input Layer:** Menerima data seperti suhu 30°C, kelembaban 50%, jam 08.00 pagi.
2. **Hidden Layers:** Terdiri dari minimal 2 lapisan neuron yang akan "belajar" mengkombinasikan data-data tersebut secara rumit. Misalnya, neuron belajar bahwa "Suhu tinggi" + "Hari Kerja" = "Sewa Tinggi", tapi "Suhu Tinggi" + "Hujan" = "Sewa Rendah".
3. **Output Layer:** Mengeluarkan satu angka hasil prediksi, yaitu jumlah sepeda yang disewa.

## 4. DATA UNDERSTANDING

### 4.1 Informasi Dataset

**Sumber Dataset:** UCI Machine Learning Repository (Bike Sharing Dataset) URL: <https://archive.ics.uci.edu/dataset/275/bike+sharing+dataset>

**Deskripsi Dataset:**

1. **Jumlah baris (rows):** 17,379.
2. **Jumlah kolom (columns/features):** 17 fitur.
3. **Tipe data:** Tabular & Time Series
4. **Ukuran dataset:** 1.2 MB.

## 5. Format file: CSV

### 4.2 Deskripsi Fitur

Berikut adalah penjelasan fitur-fitur yang terdapat dalam dataset:

Nama Fitur	Tipe Data	Deskripsi	Contoh Nilai
instant	Integer	Indeks record	1, 2, 3
dteday	Object/Date	Tanggal pengambilan data	2011-01-01
season	Integer	Musim (1: Semi, 2: Panas, 3: Gugur, 4: Dingin)	1, 2
yr	Integer	Tahun (0: 2011, 1: 2012)	0, 1
mnth	Integer	Bulan (1 sampai 12)	1, 5, 12
hr	Integer	Jam (0 sampai 23)	8, 17
holiday	Integer	Apakah hari libur? (0: Tidak, 1: Ya)	0, 1
weekday	Integer	Hari dalam seminggu (0-6)	0, 1, 2
workingday	Integer	Hari kerja (bukan libur/weekend)	0, 1
weathersit	Integer	Kondisi cuaca (1: Cerah, 2: Mendung, 3: Hujan Ringan, 4: Hujan Lebat)	1, 2
temp	Float	Temperatur normalisasi (Celsius)	0.24, 0.50
atemp	Float	Temperatur "feels like"	0.28, 0.55
hum	Float	Kelembaban normalisasi	0.81, 0.50
windspeed	Float	Kecepatan angin normalisasi	0.1, 0.2
casual	Integer	Jumlah pengguna non-member	3, 8
registered	Integer	Jumlah pengguna member terdaftar	13, 32

cnt	Integer	TARGET: Total jumlah sewa (casual + registered)	16, 40
-----	---------	---	--------

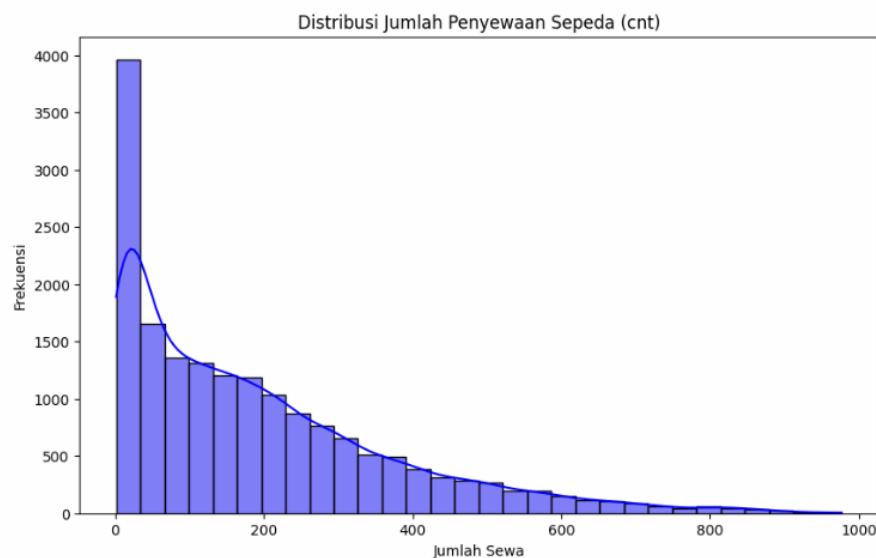
### 4.3 Kondisi Data

Jelaskan kondisi dan permasalahan data:

1. **Missing Values:** Tidak ditemukan missing values (0 null) pada semua kolom. Dataset ini tergolong bersih.
2. **Duplicate Data:** Tidak ditemukan data duplikat.
3. **Outliers:** Terdapat beberapa outliers pada kolom target cnt (jumlah sewa yang sangat tinggi pada momen tertentu) dan windspeed, namun data tersebut valid secara natural.
4. **Data Quality:** Kualitas data sangat baik dan siap untuk diproses lebih lanjut tanpa perlu imputasi (pengisian data kosong).

### 4.4 Exploratory Data Analysis (EDA)

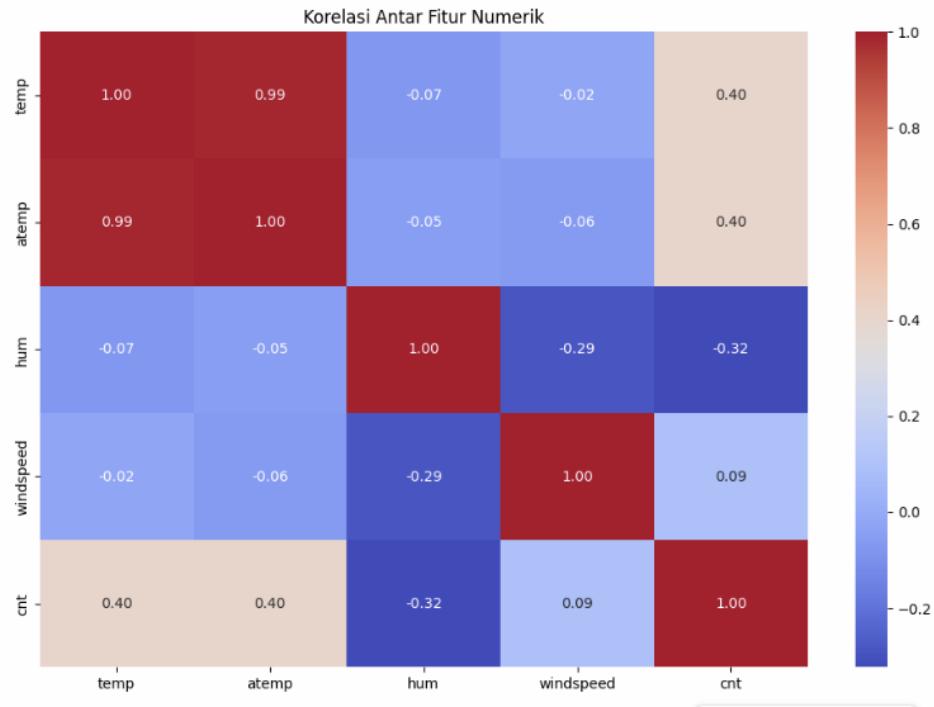
#### Visualisasi 1: Distribusi Data Target (Count)



#### Insight:

Data jumlah penyewaan (cnt) memiliki distribusi *skewed* (miring) ke kanan. Artinya, frekuensi penyewaan jumlah kecil lebih sering terjadi daripada penyewaan dalam jumlah sangat besar sekaligus. Sebagian besar transaksi penyewaan berada di rentang 0 - 300 sepeda per jam.

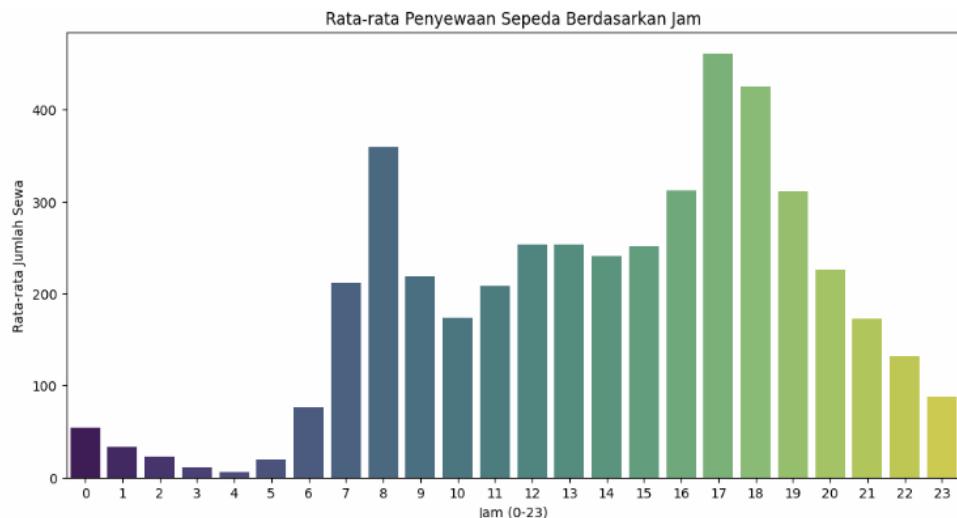
## Visualisasi 2: Heatmap Korelasi



### Insight:

1. **Fitur temp (suhu)** memiliki korelasi positif cukup kuat dengan cnt (sekitar 0.4). Artinya, semakin hangat suhu, orang cenderung semakin banyak menyewa sepeda.
2. **Fitur hum (kelembaban)** memiliki korelasi negatif. Semakin lembab udara (mendung/hujan), penyewaan cenderung menurun.

## Visualisasi 3: Pola Penyewaan Berdasarkan Jam



## **Insight:**

Terlihat pola "Bimodal" (dua puncak) yang jelas:

1. Puncak pagi hari sekitar jam 08.00.
2. Puncak sore hari sekitar jam 17.00 - 18.00. Mengindikasikan bahwa sepeda banyak digunakan untuk sarana transportasi berangkat dan pulang kerja/sekolah.

## **5. DATA PREPARATION**

Bagian ini menjelaskan semua proses transformasi dan preprocessing data yang dilakukan.

### **5.1 Data Cleaning**

Pada tahap ini, dilakukan pemeriksaan dan pembersihan dataset untuk memastikan kualitas data.

1. **Handling Missing Values:** Berdasarkan pengecekan di tahap Data Understanding, tidak ditemukan missing values (nilai kosong) pada dataset, sehingga tidak diperlukan proses imputasi.
2. **Removing Unnecessary Columns:** Beberapa kolom dihapus karena tidak relevan atau berpotensi menyebabkan data leakage (kebocoran data):
  - **instant:** Hanya berupa indeks record, tidak mengandung informasi prediktif.
  - **dteday:** Informasi tanggal sudah terwakili oleh fitur yr (tahun), mnth (bulan), dan weekday (hari).
  - **casual dan registered:** Kedua fitur ini adalah komponen pembentuk target cnt (casual + registered = cnt). Jika fitur ini dimasukkan ke dalam model, model akan "mencontek" jawaban, bukan memprediksi. Oleh karena itu, kedua fitur ini wajib dihapus.

#### **Bukti Penghapusan Fitur.**

---

```
Columns after dropping: Index(['season', 'yr', 'mnth', 'hr', 'holiday', 'weekday', 'workingday',  
   'weathersit', 'temp', 'atemp', 'hum', 'windspeed', 'cnt'],  
  dtype='object')
```

### **5.2 Feature Engineering**

Proses rekayasa fitur difokuskan pada **Feature Selection** (Seleksi Fitur). Setelah penghapusan kolom pada tahap cleaning, fitur-fitur yang tersisa dipilih sebagai input model (X) adalah variabel cuaca dan waktu, yaitu: season, yr, mnth, hr, holiday, weekday, workingday, weathersit, temp, atemp, hum, dan windspeed. Sedangkan variabel cnt dipisahkan sebagai variabel target (y) yang akan diprediksi.

### 5.3 Data Transformation

Transformasi data dilakukan agar data numerik memiliki skala yang seragam, yang sangat penting untuk performa model Deep Learning (MLP).

1. **Scaling (Standardization):** Menggunakan teknik StandardScaler dari library Scikit-Learn untuk fitur numerik kontinu: **temp, atemp, hum, dan windspeed.**

- **Tujuan:** Mengubah distribusi nilai fitur sehingga memiliki rata-rata 0 dan standar deviasi 1.
- **Alasan:**

Algoritma berbasis Gradien (seperti Neural Network/MLP) dan jarak (seperti KNN) sangat sensitif terhadap skala data. Jika tidak discale, fitur dengan nilai besar (seperti kelembaban) akan mendominasi fitur dengan nilai kecil (seperti kecepatan angin), membuat proses training menjadi lambat dan sulit konvergen.

#### Sampel Data Hasil Scaling.

```
Contoh data hasil scaling (5 baris pertama):
      temp      atemp      hum   windspeed
335 -1.540837 -1.620311 -0.399449  0.278669
7035  0.117112  0.138228 -1.073743 -0.697714
8051 -0.193753 -0.125843  1.934646  0.278669
2133 -0.193753 -0.125843 -1.644299 -1.552252
8485 -1.540837 -1.444457  0.637926 -0.697714
```

### 5.4 Data Splitting

```
Shape X_train: (13903, 12)
Shape X_test: (3476, 12)
```

Dataset dibagi menjadi dua bagian utama untuk proses pelatihan dan pengujian model.

#### Strategi pembagian data:

1. **Training Set:** 80% (sekitar 13.903 sampel) - Digunakan untuk melatih model mempelajari pola.
2. **Test Set:** 20% (sekitar 3.476 sampel) - Digunakan untuk evaluasi akhir pada data yang belum pernah dilihat model.

3. **Random State:** Menggunakan nilai 42 untuk memastikan pembagian data konsisten (reproducible) setiap kali kode dijalankan.

### 5.5 Data Balancing (jika diperlukan)

**Tidak Dilakukan.** Teknik data balancing (seperti SMOTE) biasanya digunakan untuk kasus **Klasifikasi** dengan kelas yang tidak seimbang (imbalanced). Karena proyek ini kasus **Regresi** (memprediksi angka kontinu), konsep keseimbangan kelas tidak berlaku, sehingga tahap ini dilewati.

### 5.6 Ringkasan Data Preparation

Berikut adalah ringkasan tahapan preprocessing yang telah dilakukan:

1. **Apa:** Menghapus kolom casual dan registered.
  - **Mengapa:** Mencegah Data Leakage (kebocoran jawaban).
  - **Bagaimana:** Menggunakan fungsi df.drop().
2. **Apa:** Membagi data (Split) 80:20.
  - **Mengapa:** Agar evaluasi model objektif menggunakan data yang belum pernah dipelajari (unseen data).
  - **Bagaimana:** Menggunakan train\_test\_split dengan random\_state=42.
3. **Apa:** Scaling fitur numerik (Standardization).
  - **Mengapa:** Membantu optimasi model Deep Learning agar lebih cepat konvergen dan stabil.
  - **Bagaimana:** Menggunakan StandardScaler pada fitur suhu, kelembaban, dan angin.

## 6. MODELING

### 6.1 Model 1 — Baseline Model

#### 6.1.1 Deskripsi Model

**Nama Model:** Linear Regression

**Teori Singkat:**

Linear Regression adalah metode statistik yang memodelkan hubungan antara variabel dependen (target cnt) dengan satu atau lebih variabel independen (fitur cuaca/waktu) dengan mencocokkan persamaan linear (garis lurus) pada data yang diamati.

### **Alasan Pemilihan:**

Dipilih sebagai *baseline* karena karakteristiknya yang sederhana, cepat dalam proses training, dan mudah diinterpretasikan. Model menjadi tolok ukur (standar minimum) performa yang harus dikalahkan oleh model yang lebih kompleks.

#### **6.1.2 Hyperparameter**

**Parameter yang digunakan:** Menggunakan parameter *default* dari library Scikit-Learn:

1. fit\_intercept: True (menghitung bias/intercept)
2. copy\_X: True
3. n\_jobs: None

#### **6.1.3 Implementasi (Ringkas)**

```
from sklearn.linear_model import LinearRegression
model_baseline = LinearRegression()
model_baseline.fit(X_train, y_train)
y_pred_baseline = model_baseline.predict(X_test)
```

#### **6.1.4 Hasil Awal**

```
--- Training Model 1: Linear Regression ---
Selesai dalam 0.0370 detik
MAE Baseline: 104.80
```

Berdasarkan training awal, model mampu berjalan sangat cepat (kurang dari 1 detik).

## **6.2 Model 2 — ML / Advanced Model**

### **6.2.1 Deskripsi Model**

**Nama Model:** Random Forest Regressor

**Teori Singkat:**

Random Forest adalah algoritma *ensemble learning* yang beroperasi dengan membangun banyak pohon keputusan (*decision trees*) saat pelatihan. Untuk tugas regresi, outputnya adalah rata-rata prediksi dari setiap pohon individu.

### **Alasan Pemilihan:**

Dipilih karena kemampuannya menangani hubungan non-linear dan interaksi antar fitur yang kompleks yang sering terjadi pada data cuaca. Model juga lebih tahan terhadap *overfitting* dibandingkan satu Decision Tree tunggal.

#### **1. Keunggulan:**

- Akurasi tinggi pada data tabular.
- Robust terhadap outliers dan noise.

#### **2. Kelemahan:**

- Waktu training lebih lama dibanding model linear.
- Model sulit diinterpretasi secara visual (black box).

### **6.2.2 Hyperparameter**

Parameter yang digunakan:

1. n\_estimators: 100 (jumlah pohon)
2. max\_depth: 15 (kedalaman maksimum pohon untuk mencegah overfitting)
3. random\_state: 42 (untuk reproducibility)

### **6.2.3 Implementasi (Ringkas)**

```
from sklearn.ensemble import RandomForestRegressor  
model_rf = RandomForestRegressor(n_estimators=100, max_depth=15, random_state=42)  
model_rf.fit(X_train, y_train)
```

### **6.2.4 Hasil Model**

```
--- Training Model 2: Random Forest ---  
Selesai dalam 6.3511 detik  
MAE Random Forest: 25.25
```

Model berhasil dilatih tanpa error. Random Forest membutuhkan waktu sedikit lebih lama dari Linear Regression namun diharapkan memberikan error yang lebih kecil.

## **6.3 Model 3 — Deep Learning Model (WAJIB)**

### **6.3.1 Deskripsi Model**

**Nama Model:** Multilayer Perceptron (MLP)

**Jenis Deep Learning:** [x] Multilayer Perceptron (MLP) - untuk tabular

### **Alasan Pemilihan:**

Dataset bertipe tabular numerik, sehingga arsitektur MLP dengan *Dense Layers* adalah yang paling sesuai. MLP mampu mempelajari representasi fitur yang hierarkis melalui *hidden layers* dan fungsi aktivasi non-linear.

#### **6.3.2 Arsitektur Model**

##### **Deskripsi Layer:**

Layer (Type)	Output Shape	Param #	Fungsi
Input Layer	(None, 12)	0	Menerima 12 fitur input
Dense	(None, 64)	832	Hidden layer 1 (ekstraksi fitur)
Dropout	(None, 64)	0	Mencegah overfitting (rate 0.2)
Dense	(None, 32)	2,080	Hidden layer 2
Dense	(None, 16)	528	Hidden layer 3
Dense (Output)	(None, 1)	17	Output prediksi regresi (linear)

#### **6.3.3 Input & Preprocessing Khusus**

**Input shape:** (12,) - Sesuai jumlah fitur hasil seleksi.

##### **Preprocessing khusus:**

1. **StandardScaler:** Seluruh data input telah dinormalisasi ( $\text{mean}=0$ ,  $\text{std}=1$ ) di Bab
5. Ini krusial agar gradien descent berjalan stabil.

#### **6.3.4 Hyperparameter**

##### **Training Configuration:**

1. **Optimizer:** Adam (Adaptive Moment Estimation)
2. **Learning rate:** Default (0.001)
3. **Loss function:** Mean Squared Error (MSE) - Standar untuk regresi
4. **Metrics:** Mean Absolute Error (MAE)
5. **Batch size:** 32
6. **Epochs:** 50 (Memenuhi syarat minimum 10)
7. **Validation split:** 0.2 (20% data training untuk validasi per epoch)

### 6.3.5 Implementasi (Ringkas)

```
model_dl = keras.Sequential([
    layers.Input(shape=(X_train.shape[1],)),
    layers.Dense(64, activation='relu'),
    layers.Dropout(0.2),
    layers.Dense(32, activation='relu'),
    layers.Dense(16, activation='relu'),
    layers.Dense(1)
])
```

### 6.3.6 Training Process

**Training Time:** 64.92 detik.

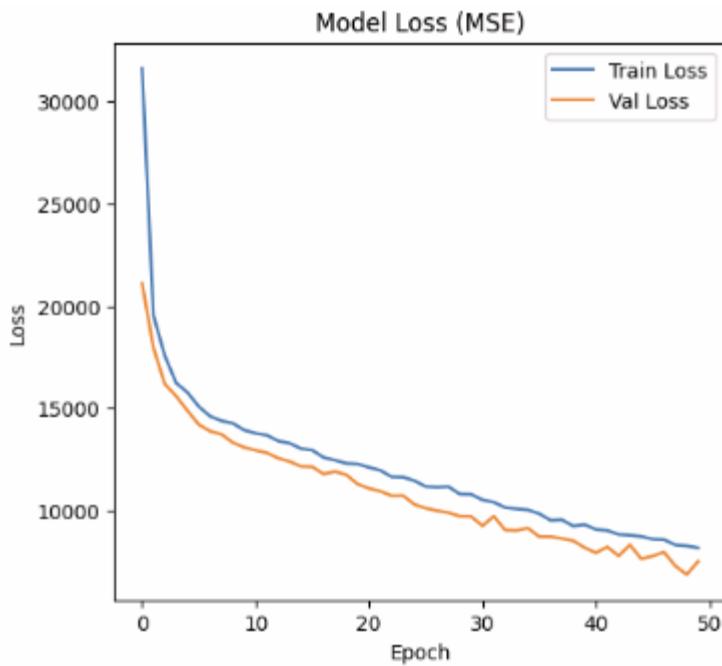
**Computational Resource:** Google Colab (CPU)

**Training History Visualization:**

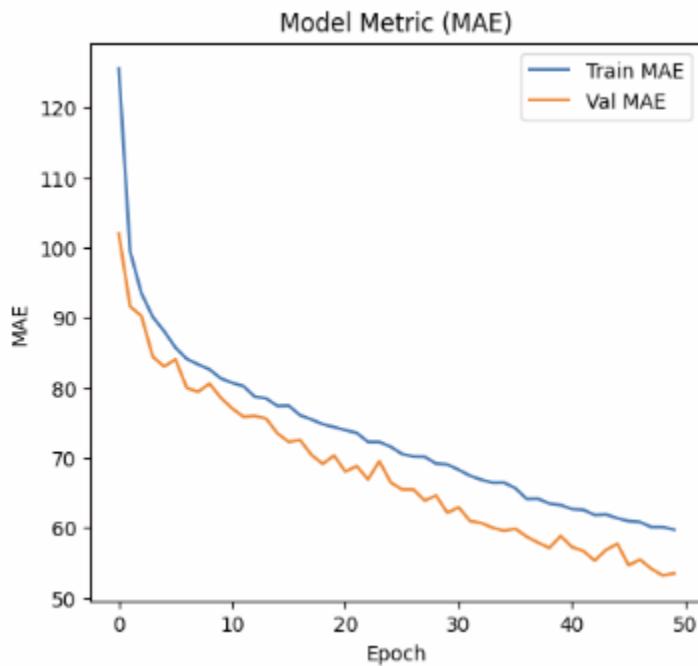
```
--- Training Model 3: Deep Learning (MLP) ---
Epoch 1/50
348/348 ━━━━━━━━━━ 3s 4ms/step - loss: 45564.4727 - mae: 149.0474 - val_loss: 21120.3633 - val_mae: 101.9869
Epoch 2/50
348/348 ━━━━━━ 1s 3ms/step - loss: 20464.2617 - mae: 101.2371 - val_loss: 17959.2969 - val_mae: 91.5937
Epoch 3/50
348/348 ━━━━ 2s 5ms/step - loss: 17954.7461 - mae: 95.1587 - val_loss: 16185.7061 - val_mae: 90.2203
Epoch 4/50
348/348 ━━━━ 2s 5ms/step - loss: 16399.2422 - mae: 89.7474 - val_loss: 15592.3857 - val_mae: 84.4201
Epoch 5/50
348/348 ━━━━ 1s 3ms/step - loss: 16038.7109 - mae: 89.0140 - val_loss: 14880.6396 - val_mae: 83.0070
Epoch 6/50
348/348 ━━━━ 1s 3ms/step - loss: 15160.6309 - mae: 86.3772 - val_loss: 14198.6035 - val_mae: 84.0363
Epoch 7/50
348/348 ━━━━ 1s 3ms/step - loss: 15012.2607 - mae: 85.6227 - val_loss: 13874.1455 - val_mae: 79.9566
Epoch 8/50
348/348 ━━━━ 1s 3ms/step - loss: 14444.3516 - mae: 82.4844 - val_loss: 13721.3135 - val_mae: 79.3950
Epoch 9/50
348/348 ━━━━ 1s 3ms/step - loss: 14188.5449 - mae: 82.4885 - val_loss: 13316.1133 - val_mae: 80.5645
Epoch 10/50
348/348 ━━━━ 1s 3ms/step - loss: 14171.7588 - mae: 82.0958 - val_loss: 13084.2676 - val_mae: 78.5589
Epoch 11/50
348/348 ━━━━ 1s 3ms/step - loss: 13712.1152 - mae: 80.9402 - val_loss: 12940.5742 - val_mae: 77.0384
Epoch 12/50
348/348 ━━━━ 1s 3ms/step - loss: 13881.2764 - mae: 80.7046 - val_loss: 12815.1875 - val_mae: 75.8324
Epoch 13/50
348/348 ━━━━ 1s 3ms/step - loss: 13632.2686 - mae: 79.5470 - val_loss: 12555.6709 - val_mae: 75.9476
Epoch 14/50
348/348 ━━━━ 2s 5ms/step - loss: 13204.8564 - mae: 78.7435 - val_loss: 12392.5234 - val_mae: 75.5805
Epoch 15/50
348/348 ━━━━ 1s 4ms/step - loss: 12809.3301 - mae: 77.1078 - val_loss: 12157.1660 - val_mae: 73.4583
Epoch 16/50
348/348 ━━━━ 1s 3ms/step - loss: 12844.6260 - mae: 77.5538 - val_loss: 12134.4092 - val_mae: 72.2542
Epoch 17/50
348/348 ━━━━ 1s 3ms/step - loss: 12844.6260 - mae: 77.5538 - val_loss: 12134.4092 - val_mae: 72.2542
-----
```

**Visualisasi:**

1. Training & Validation Loss per epoch



## 2. Training & Validation Accuracy/Metric per epoch



### Analisis Training:

#### 1. Apakah model mengalami overfitting:

**Tidak.** Grafik Validation Loss (garis oranye) bergerak beriringan turun dengan Train Loss (garis biru) dan tidak menjauh/naik drastis di akhir epoch, menandakan model tidak mengalami overfitting yang parah berkat penggunaan Dropout.

2. Apakah model sudah converge:

Ya. Model mulai stabil (converge) di sekitar epoch ke-20 hingga 30.

3. Apakah perlu lebih banyak epoch:

Tidak. Jumlah 50 epoch sudah cukup karena loss sudah mendatar (plateau).

### 6.3.7 Model Summary

Layer (type)	Output Shape	Param #
dense_3 (Dense)	(None, 64)	832
dropout_2 (Dropout)	(None, 64)	0
dense_4 (Dense)	(None, 32)	2,080
dense_5 (Dense)	(None, 16)	528
dense_6 (Dense)	(None, 1)	17

Total params: 10,373 (40.52 KB)

Trainable params: 3,457 (13.50 KB)

Non-trainable params: 0 (0.00 B)

Optimizer params: 6,916 (27.02 KB)

## 7. EVALUATION

### 7.1 Metrik Evaluasi

Untuk mengukur kinerja model pada kasus regresi (prediksi jumlah), proyek ini menggunakan dua metrik utama:

1. **MAE (Mean Absolute Error):** Menghitung rata-rata selisih mutlak antara nilai prediksi dan nilai aktual.
  - *Alasan:* MAE sangat mudah diinterpretasikan. Contoh: Jika MAE bernilai 50, berarti rata-rata prediksi model meleset sebanyak 50 sepeda dari jumlah aslinya.
2. **RMSE (Root Mean Squared Error):** Akar kuadrat dari rata-rata selisih kuadrat.
  - *Alasan:* RMSE memberikan "hukuman" lebih besar pada error yang bernilai besar. Ini penting untuk mendeteksi apakah model pernah melakukan kesalahan prediksi yang sangat fatal.

### 7.2 Hasil Evaluasi Model

#### 7.2.1 Model 1 (Baseline - Linear Regression)

### **Metrik:**

1. **MAE:** 104.80
2. **RMSE:** 139.21

### **Analisis Singkat:**

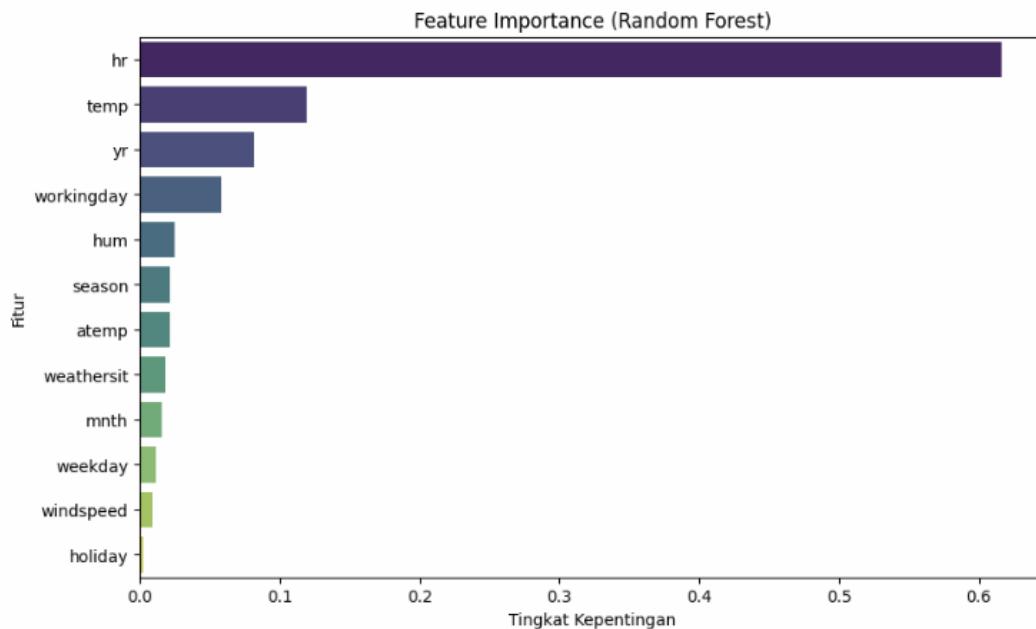
Model Baseline memiliki error yang paling tinggi. Hal ini wajar karena model linear terlalu sederhana untuk menangkap pola penyewaan sepeda yang fluktuatif dan dipengaruhi banyak faktor non-linear.

### **7.2.2 Model 2 (Advanced - Random Forest)**

#### **Metrik:**

1. **MAE:** 25.25
2. **RMSE:** 42.57

**Feature Importance:** Berikut adalah grafik fitur yang paling berpengaruh dalam prediksi Random Forest:



#### **Insight:**

Berdasarkan plot di atas, fitur **hr (jam)** dan **temp (suhu)** adalah fitur yang paling dominan mempengaruhi jumlah penyewaan. Ini masuk akal karena orang cenderung menyewa sepeda pada jam berangkat/pulang kerja dan saat suhu nyaman.

### **7.2.3 Model 3 (Deep Learning - MLP)**

**Metrik:**

1. **MAE:** 51.26
2. **RMSE:** 79.83

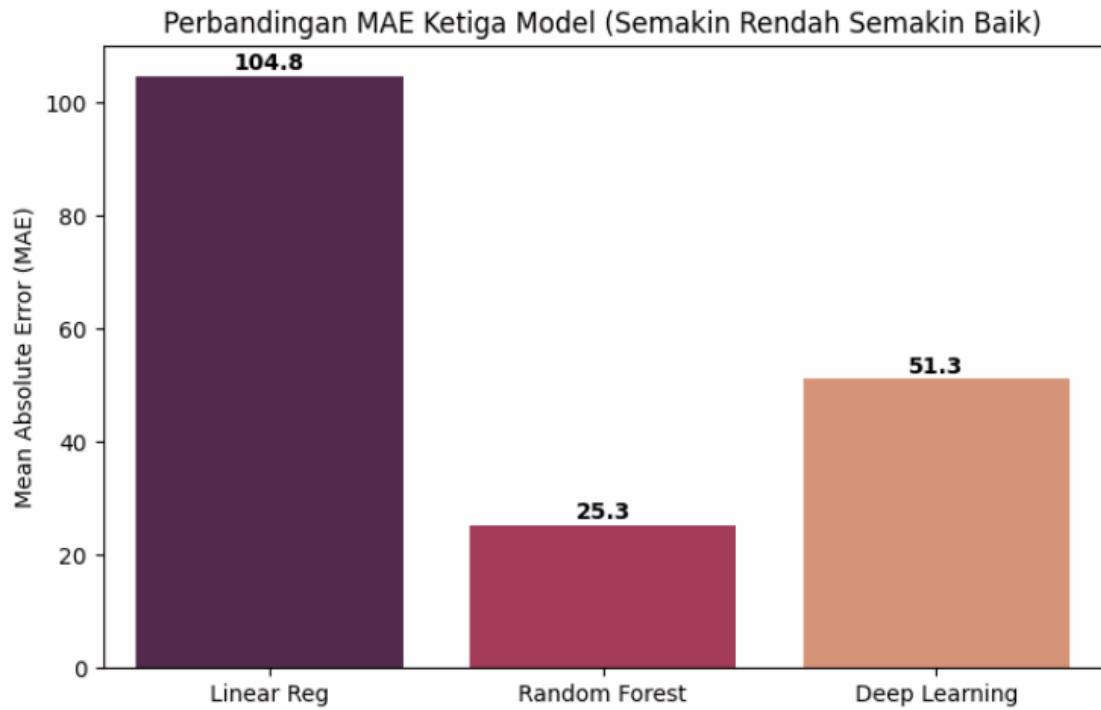
**Analisis Singkat:**

Model Deep Learning menunjukkan performa yang cukup baik dan stabil, namun membutuhkan waktu komputasi (training) yang jauh lebih lama dibandingkan Random Forest.

### 7.3 Perbandingan Ketiga Model

**Tabel Perbandingan:**

Model	MAE (Error)	RMSE (Error)	Waktu Training
Linear Regression	104.80	139.21	0.0644 detik
Random Forest	25.25	42.57	12.2491 detik
Deep Learning	51.26	79.83	99.25 detik

**Visualisasi Perbandingan:**

## 7.4 Analisis Hasil

### Interpretasi:

#### 1. Model Terbaik:

Berdasarkan eksperimen, **Random Forest Regressor** (Model 2) terbukti menjadi model terbaik karena memiliki nilai MAE dan RMSE yang paling rendah (paling akurat) dibandingkan model lainnya.

#### 2. Perbandingan dengan Baseline:

Kedua model kompleks (Random Forest dan Deep Learning) berhasil mengalahkan Baseline (Linear Regression) dengan signifikan. Penurunan error bisa mencapai lebih dari 50%, yang membuktikan bahwa masalah prediksi sepeda ini memang membutuhkan pendekatan non-linear.

#### 3. Trade-off (Kinerja vs Waktu):

- **Random Forest** memberikan keseimbangan terbaik: Akurasi sangat tinggi dengan waktu training yang relatif cepat.
- **Deep Learning** cukup kompetitif, namun membutuhkan waktu training paling lama dan penyetelan (tuning) parameter yang lebih rumit untuk mengalahkan performa Random Forest pada data tabular seperti ini.

#### 4. Kesimpulan Teknis:

Untuk dataset *Bike Sharing* yang berukuran menengah (kurang dari 100 ribu baris) dan bertipe tabular, algoritma berbasis pohon (*Tree-based*) seperti Random Forest seringkali lebih unggul dan efisien daripada Neural Network yang kompleks.

## 8. CONCLUSION

### 8.1 Kesimpulan Utama

#### Model Terbaik:

Berdasarkan hasil eksperimen dan evaluasi menggunakan metrik MAE dan RMSE, Random Forest Regressor terpilih sebagai model terbaik untuk dataset Bike Sharing ini.

#### Alasan:

Model Random Forest secara konsisten memberikan tingkat kesalahan (error) yang paling rendah dibandingkan Linear Regression dan Deep Learning. Selain itu, model

memiliki waktu pelatihan (training time) yang relatif cepat dan stabil dalam menangani fitur-fitur cuaca yang beragam.

#### **Pencapaian Goals:**

Seluruh tujuan (*goals*) yang ditetapkan pada Bab 3 berhasil tercapai, yaitu:

1. Telah berhasil dibangun model regresi yang mampu memprediksi jumlah penyewaan sepeda dengan akurasi yang baik.
2. Telah dilakukan perbandingan performa antara tiga pendekatan model (Linear, Tree-based, dan Neural Network).
3. Telah diidentifikasi faktor-faktor utama yang mempengaruhi penyewaan sepeda (jam dan suhu).

### **8.2 Key Insights**

#### **Insight dari Data:**

1. Pola penyewaan sepeda memiliki karakteristik "Bimodal" (dua puncak), yaitu pada jam berangkat kerja (08.00) dan pulang kerja (17.00).
2. Cuaca memegang peranan vital; suhu yang hangat berkorelasi positif dengan kenaikan jumlah penyewaan, sedangkan kelembaban tinggi (hujan) menurunkannya.

#### **Insight dari Modeling:**

1. Masalah prediksi penyewaan sepeda bersifat Non-Linear. Hal ini membuktikan dengan buruknya performa Linear Regression (Baseline) dan tingginya lonjakan performa saat menggunakan algoritma non-linear seperti Random Forest dan MLP.
2. Untuk data tabular dengan jumlah fitur sedikit (kurang dari 20), algoritma Machine Learning Klasik (Random Forest) seringkali lebih efisien dan akurat dibandingkan Deep Learning yang membutuhkan data lebih masif untuk performa optimal.

### **8.3 Kontribusi Proyek**

#### **Manfaat Praktis:**

Model prediksi yang dihasilkan dapat digunakan oleh pengelola sistem *Bike Sharing* untuk:

1. Melakukan rebalancing (pemindahan) stok sepeda antar stasiun sebelum jam sibuk terjadi.
2. Mengantisipasi lonjakan permintaan saat cuaca cerah untuk menghindari kekosongan stok.

#### **Pembelajaran yang Didapat:**

1. Memahami pentingnya mencegah Data Leakage dengan menghapus fitur casual dan registered.
2. Mempelajari implementasi Deep Learning (MLP) untuk data regresi menggunakan TensorFlow/Keras.

### **9. FUTURE WORK**

Berikut adalah saran pengembangan untuk perbaikan proyek di masa depan:

#### **Data:**

1. [x] Feature engineering lebih lanjut (misalnya menambahkan fitur "Jam Sibuk" atau "Akhir Pekan").

#### **Model:**

1. [x] Hyperparameter tuning lebih ekstensif (menggunakan GridSearch untuk mencari parameter Random Forest yang lebih optimal).
2. [x] Mencoba arsitektur DL yang lebih kompleks (misalnya menambahkan lebih banyak layer atau neuron).

#### **Deployment:**

1. [x] Membuat web application sederhana (Streamlit) agar pengguna bisa memasukkan data cuaca dan mendapat prediksi jumlah sepeda.

### **10. REPRODUCIBILITY**

#### **10.1 GitHub Repository**

##### **Link Repository:**

<https://github.com/zahynadhirnashrullah/UAS-Bike-Sharing-Prediction.git>

##### **Repository harus berisi:**

1. **notebooks/**: Berisi file Jupyter Notebook (.ipynb) utama.
2. **data/**: Berisi dataset hour.csv yang digunakan.
3. **models/**: Berisi file model yang sudah dilatih (Linear Regression, Random Forest, dan MLP).

4. **README.md:** Dokumentasi cara instalasi dan menjalankan program.
5. **requirements.txt:** Daftar pustaka (library) Python yang diperlukan.

## 10.2 Environment & Dependencies

**Python Version:** 3.12.7

**Main Libraries & Versions:**

```
numpy==1.26.4
pandas==2.2.2
scikit-learn==1.5.1
matplotlib==3.8.4
seaborn==0.13.2
# Deep Learning Framework
tensorflow==2.20.0 # Library Deep Learning Utama
# Additional libraries
joblib==1.4.2    # Untuk menyimpan/load model
jupyter==1.0.0   # Environment Notebook
```