

Studi Kasus: Analisis Indeks Pekerja Tetap Konstruksi Indonesia Tahun 2024

NIM : A11.2023.15180

Nama : Muhammad Fikri Alif Karim

1. Business Understanding

Latar Belakang:

Sektor konstruksi merupakan salah satu penopang pembangunan ekonomi di Indonesia. Jumlah pekerja tetap konstruksi yang stabil dan meningkat dapat mencerminkan kondisi pertumbuhan ekonomi dan pembangunan infrastruktur di suatu provinsi.

Masalah:

- Pemerintah ingin memahami tren tenaga kerja tetap sektor konstruksi di berbagai provinsi.
- Data indeks pekerja tetap ini digunakan untuk memantau produktivitas tenaga kerja konstruksi dan memastikan ketersediaan SDM dalam proyek infrastruktur.

Tujuan:

- Menganalisis indeks triwulanan pekerja tetap konstruksi di seluruh provinsi pada tahun 2024.
- Mengidentifikasi provinsi dengan indeks tertinggi dan terendah.
- Memberikan dasar rekomendasi kebijakan terkait distribusi dan penguatan tenaga kerja konstruksi.

2. Data Understanding

Sumber data: <https://www.bps.go.id/id/statistics-table/2/NTQ4IzI=/indeks-triwulanan-pekerja-tetap-konstruksi-menurut-provinsi--2016-100-.html>

- Atribut:
 1. Provinsi
 2. Triwulan (I, II, III, IV tahun 2024)
 3. Indeks Pekerja Tetap Konstruksi (2016=100)

Tipe Data:

- Provinsi : Nominal (kategori)
- Triwulan : Ordinal (urutan 1–4)
- Indeks : Numerik (rasio, skala relatif 2016=100)

Wawasan Awal:

- Indeks di atas 100 menunjukkan peningkatan jumlah pekerja tetap dibanding tahun 2016.
- Beberapa provinsi dengan pembangunan pesat seperti DKI Jakarta, Jawa Barat, dan Kalimantan Timur kemungkinan memiliki indeks lebih tinggi.
- Provinsi di kawasan timur cenderung memiliki indeks lebih rendah karena pembangunan infrastruktur relatif lebih lambat.

3. Data Preparation

Langkah persiapan data:

- Seleksi Data: Fokus pada atribut Provinsi, Triwulan, dan Indeks.
- Pembersihan Data: Memastikan tidak ada nilai kosong pada indeks.
- Transformasi:
 - Normalisasi indeks agar memudahkan perbandingan antarprovinsi.
 - Penambahan atribut “Kategori Pertumbuhan” dengan kriteria Rendah (<95), Sedang ($95-105$), Tinggi (>105).
- Agregasi:
 - Menghitung rata-rata indeks tiap provinsi sepanjang 2024.
 - Menghitung tren triwulanan nasional untuk melihat kenaikan atau penurunan.

4. Modeling

Pada tahap ini, metode **klasifikasi** dipilih untuk memodelkan data. Teknik yang digunakan adalah **Decision Tree** karena cocok untuk data kategori seperti provinsi, triwulan, dan kelas pertumbuhan.

Langkah-langkah:

1. **Target Variable (Label):**
 - Kategori Pertumbuhan Indeks (Rendah, Sedang, Tinggi).
 - Rendah jika indeks < 95 .
 - Sedang jika $95 \leq \text{indeks} \leq 105$.
 - Tinggi jika indeks > 105 .

2. Fitur (Atribut Input):

- Provinsi
- Triwulan (I, II, III, IV)
- Nilai indeks (numerik, kemudian dikategorikan atau digunakan langsung untuk pembentukan aturan).

3. Proses Pemodelan:

- Dataset dibagi menjadi data latih dan data uji (misalnya 80% latih, 20% uji).
- Algoritma Decision Tree digunakan untuk membentuk aturan klasifikasi.
- Aturan berbentuk pohon keputusan dengan kondisi *if-then*.

4. Contoh Aturan yang Dihasilkan (hipotetis):

- Jika indeks > 105 maka kategori = Tinggi.
- Jika indeks antara 95–105 dan provinsi = Jawa Barat maka kategori = Sedang.
- Jika indeks < 95 dan provinsi = Papua maka kategori = Rendah.

5. Output Model:

- Pohon keputusan yang memetakan setiap provinsi dan triwulan ke kategori pertumbuhan.
- Dapat digunakan untuk memprediksi kategori pertumbuhan provinsi di triwulan selanjutnya.

5. Evaluation

Metrik evaluasi:

- Confusion Matrix untuk melihat distribusi prediksi benar dan salah.
- Accuracy untuk mengukur tingkat ketepatan keseluruhan model.
- Precision dan Recall untuk mengukur ketepatan serta sensitivitas klasifikasi pada setiap kategori.

Hasil evaluasi (contoh hipotetis):

- Akurasi model sebesar 87 persen.
- Precision: Tinggi (0,89), Sedang (0,84), Rendah (0,86).
- Recall: Tinggi (0,91), Sedang (0,80), Rendah (0,85).

Interpretasi:

- Model mampu mengidentifikasi provinsi dengan kategori pertumbuhan tinggi secara baik.
- Kategori sedang relatif lebih sulit diprediksi karena adanya provinsi yang kadang bergeser ke kategori rendah atau tinggi akibat fluktuasi.
- Secara keseluruhan, model cukup handal untuk memberikan gambaran klasifikasi pertumbuhan pekerja tetap konstruksi antar provinsi.

6. Deployment

Rencana penerapan:

- Versi sederhana berupa laporan tahunan mengenai kondisi pekerja tetap konstruksi per provinsi.
- Versi lanjutan berupa dashboard interaktif yang menampilkan:
 - Tren triwulanan per provinsi.
 - Peta interaktif dengan kategori indeks (rendah, sedang, tinggi).
 - Prediksi indeks untuk periode berikutnya.

Pemeliharaan:

- Model diperbarui setiap tahun dengan data terbaru dari BPS.
- Jika terdapat proyek besar atau perubahan pola pembangunan, analisis clustering dan klasifikasi dapat diulang untuk memastikan hasil tetap relevan.

1. Upload Dataset

- Kode menggunakan `files.upload()` dari `google.colab`.
- Setelah file CSV diupload, disimpan ke `csv_path`, lalu dibaca dengan `pd.read_csv`.
- Hasil `df.head()` menunjukkan tabel awal dari BPS, masih dalam format wide (kolom Triwulan I–IV terpisah, provinsi ada di kolom pertama).

```
[12] import pandas as pd
✓ 10s from google.colab import files

# Upload file CSV
uploaded = files.upload()

# Ambil nama file yang diupload
csv_path = list(uploaded.keys())[0]
print("File berhasil diupload:", csv_path)

# Baca dataset
df = pd.read_csv(csv_path)
df.head()
```

Indeks Triw...0). 2024.csv
Indeks Triwulanan Pekerja Tetap Konstruksi Menurut Provinsi (2016–100), 2024.csv (text/csv) - 1915 bytes, last modified: 9/30/2025 - 100% done
Saving Indeks Triwulanan Pekerja Tetap Konstruksi Menurut Provinsi (2016–100), 2024.csv to Indeks Triwulanan Pekerja Tetap Konstruksi Menurut Provinsi (2016–100), 2024 (1).csv
File berhasil diupload: Indeks Triwulanan Pekerja Tetap Konstruksi Menurut Provinsi (2016–100), 2024 (1).csv

	38 Provinsi	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5
0	NaN	Indeks Triwulanan Pekerja Tetap Konstruksi Men...	NaN	NaN	NaN	NaN
1	NaN	2024	NaN	NaN	NaN	NaN
2	NaN	Triwulan I	Triwulan II	Triwulan III	Triwulan IV	Tahunan
3	ACEH	106.14	106.42	107.33	108.1	107
4	SUMATERA UTARA	111.87	115.14	116.21	117.31	115.13

2. Data Preparation

- Rename kolom jadi lebih mudah (Provinsi, Triwulan I, dst).
- Drop kolom kosong Unnamed: 5.
- `pd.melt(...)` → ubah dari wide menjadi long format:
 - Sebelum: 1 baris per provinsi, kolom Triwulan I–IV.
 - Sesudah: 1 baris per provinsi *per triwulan*.
- Pastikan kolom Indeks numerik.
- Buat kolom baru Kategori berdasarkan nilai Indeks:
 - <95 = Rendah
 - $95\text{--}105$ = Sedang
 - >105 = Tinggi
- Output: tabel work dengan kolom [Provinsi, Triwulan, Indeks, Kategori].
- Print distribusi kategori: mayoritas Tinggi, sedikit Sedang, hampir tidak ada Rendah.

```

13) # Rename kolom agar lebih mudah dipakai
df = df.rename(columns=[
    '18 Provinsi': 'Provinsi',
    'Unnamed: 1': 'Triwulan I',
    'Unnamed: 2': 'Triwulan II',
    'Unnamed: 3': 'Triwulan III',
    'Unnamed: 4': 'Triwulan IV'
])

# Buang kalau ada kolom kosong terakhir
if 'Unnamed: 5' in df.columns:
    df = df.drop(columns=['Unnamed: 5'])

# Ubah dari wide ke long format
work = df.melt(id_vars=['Provinsi'],
               value_vars=['Triwulan I', 'Triwulan II', 'Triwulan III', 'Triwulan IV'],
               var_name='Triwulan', value_name='Indeks')

# Pastikan numeric
work['Indeks'] = pd.to_numeric(work['Indeks'], errors='coerce')

# Buat kategori target
def kategori(x):
    if x < 95:
        return 'Rendah'
    elif x <= 105:
        return 'Sedang'
    else:
        return 'Tinggi'

work['Kategori'] = work['Indeks'].apply(kategori)

print(work.head())
print("\nJumlah data:", len(work))
print("\nDistribusi kategori:\n", work['Kategori'].value_counts())

```

	Provinsi	Triwulan	Indeks	Kategori
0	NaN	Triwulan I	NaN	Tinggi
1	NaN	Triwulan I	2024.00	Tinggi
2	NaN	Triwulan I	NaN	Tinggi
3	ACEH	Triwulan I	106.14	Tinggi
4	SUMATERA UTARA	Triwulan I	111.87	Tinggi

Jumlah data: 168

Distribusi kategori:

Kategori	Count
Tinggi	162
Sedang	6

Name: count, dtype: int64

3. Training Decision Tree

- Proses training:
 - OneHotEncoder → ubah kategori (Provinsi, Triwulan) ke bentuk numerik.
 - train_test_split → bagi data latih & uji (80:20).
 - DecisionTreeClassifier(max_depth=5) → melatih pohon keputusan dengan kedalaman maksimal 5.
- Output: “Model selesai dilatih”.

```

[17]
✓ Os
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.tree import DecisionTreeClassifier, plot_tree
import matplotlib.pyplot as plt

X = work[['Provinsi', 'Triwulan']]
y = work['Kategori']

# One-hot encoding
ct = ColumnTransformer([('ohe', OneHotEncoder(handle_unknown='ignore'), ['Provinsi', 'Triwulan'])])
X_enc = ct.fit_transform(X)

# Split data
X_train, X_test, y_train, y_test = train_test_split(X_enc, y, test_size=0.2, random_state=42, stratify=y)

# Decision Tree
clf = DecisionTreeClassifier(max_depth=5, random_state=42)
clf.fit(X_train, y_train)

print("Model selesai dilatih")

```

Model selesai dilatih

4. Evaluasi Model (Decision Tree)

- Import metric: `accuracy_score`, `classification_report`, `confusion_matrix`.
- `y_pred = clf.predict(X_test)` → model memprediksi data uji.
- Print akurasi: 97%.
- Classification Report menunjukkan:
 - Sedang: precision, recall = 0 (model gagal memprediksi kelas ini).
 - Tinggi: precision 0.97, recall 1.0 (model sangat bagus di kelas ini).
- Confusion Matrix divisualisasikan:
 - Hampir semua prediksi masuk kelas Tinggi.
 - Ada 1 data “Sedang” salah diprediksi.
- Kesimpulan: model bias ke kelas mayoritas.

```
[15] ✓ On
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
import numpy as np

# Prediksi data uji
y_pred = clf.predict(X_test)

# Akurasi & laporan klasifikasi
print("Akurasi:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred, labels=['Rendah', 'Sedang', 'Tinggi'])
print("Confusion Matrix:\n", cm)

# Visualisasi Confusion Matrix
plt.imshow(cm, cmap="Blues")
plt.xticks([0,1,2], ['Rendah', 'Sedang', 'Tinggi'])
plt.yticks([0,1,2], ['Rendah', 'Sedang', 'Tinggi'])
plt.xlabel("Predicted")
plt.ylabel("True")

# Tampilkan angka di dalam matrix
for i in range(cm.shape[0]):
    for j in range(cm.shape[1]):
        plt.text(j, i, cm[i, j], ha="center", va="center", color="black")

plt.show()
```

Akurasi: 0.9705882352941176

Classification Report:				
	precision	recall	f1-score	support
Sedang	0.00	0.00	0.00	1
Tinggi	0.97	1.00	0.99	33
accuracy			0.97	34
macro avg	0.49	0.50	0.49	34
weighted avg	0.94	0.97	0.96	34

5. Visualisasi Decision Tree

- Pohon keputusan divisualisasikan dengan `plot_tree`.
- Node utama memisahkan berdasarkan **provinsi tertentu** (misalnya Sulawesi Tenggara, Sulawesi Tengah, Sulawesi Utara).
- Hampir semua cabang jatuh ke kelas Tinggi (warna biru).
- Ada sedikit cabang ke kelas Sedang (warna oranye), tapi sangat jarang.
- Ini menunjukkan bahwa model hanya mengenali pola kelas Tinggi, tidak seimbang.

