Dasar Pemrosesan Data

Mohammad Febryan Khamim

Pre-processing data merupakan langkah awal yang sangat penting dalam proses analisis data dan *data science*.

1 Pengenalan Data

1.1 Tipe-Tipe Data

Terdapat beberapa tipe-tipe data berdasarkan beberapa kriteria tertentu sebagai berikut.

1. **Berdasarkan Statistika**

   Berdasakan statistika, terdapat dua jenis data yaitu Kategorikal dan Numerikal.

   - **Numerik**. Tipe data berupa angka yang dapat dilakukan operasi matematika. Jenis data Numerikal terdapat 2, yaitu interval dan rasio. **Interval** adalah data yang tak punya nol sejati, contohnya ada pada suhu dan tahun. Sementara itu, **rasio** adalah jenis data numerikal yang punya nilai nol sejati, contohnya adalah umur dan berat.
   - **Kategorikal**. Jenis data ini berupa kategori atau label. Terdapat dua jenis dalam data kategorikal, yaitu **ordinal** yang memiliki hubungan urutan atau tingkatan, misalkan *rating*. Sementara itu, terdapat pula tipe **nominal** yang tidak memiliki urutan, seperti jenis kelamin, warna.

2. **Berdasarkan Struktur**

   Berdasarkan struktur data, terdapat beberapa jenis data, yaitu *structured*, *semi-structured*, dan *unstructured* data.

   - **Structured**. Merupakan data yang sudah rapi dalam bentuk baris dan kolom.
   - **Semi-structured**. Merupakan data yang tak memiliki tabel, tetapi mempunyai pola atau label.
   - **Unstructured**. Data yang tak memiliki format tetap.

2 Handling Missing Values

Data yang hilang atau *missing values* adalah suatu kondisi ketika beberapa entri data yang kosong, bernilai NaN, None, atau Special String seperti "*unknown*".
Kenapa harus ditangani? Karena *missing values* dapat mengakibatkan beberapa hal negatif, di antaranya mengurangi akurasi, bias, dan merusak algoritma.

Terdapat beberapa jenis *missing values*, di antaranya sebagai berikut.

- **MCAR (Missing Completely at Random)**
  Nilai hilang terjadi secara acak dan tidak dipengaruhi oleh nilai variabel lain maupun nilai variabel itu sendiri. Kehilangan data tidak memiliki pola tertentu.
- **MAR (Missing at Random)**
  Nilai hilang tidak bergantung pada nilai yang hilang itu sendiri, tetapi bergantung pada variabel lain. Ada pola tertentu yang dapat dijelaskan dengan variabel lain dalam data.
- **MNAR (Missing Not at Random)**
  Nilai hilang bergantung pada nilai yang hilang itu sendiri. Ketidakhadiran data memiliki alasan khusus, misalnya responden sengaja tidak menjawab karena nilai terlalu sensitif atau ekstrem.

Untuk menangani nilai atau data yang hilang, terdapat beberapa teknik yang dapat digunakan, di antaranya sebagai berikut.

1. **Hapus baris dengan *missing values*** → cocok untuk MCAR.
   - ⊕ Simple, cepat, praktis.
   - ⊖ Kehilangan data & bias.
2. **Mengisi dengan *mean, median, atau modus***.
   - ⊕ Mudah, cepat, cocok untuk data numerik.
   - ⊖ Mengubah distribusi data, merusak varians.

   Catatan: masing-masing mean, median, modus punya (+/-)
   - **Mean**:
     - ⊕ Mempertahankan mean.
     - ⊖ Merusak varians, korelasi antar variabel berubah, data menjadi terlalu rata.
   - **Median**:
     - ⊕ Lebih robust terhadap *outlier*, tak terpengaruh ekstrem.
     - ⊖ Mengurangi variabilitas, bersifat kasar.
   - **Modus**:
     - ⊕ Simpel.
     - ⊖ Membuat dominan makin dominan.
3. ***Forward and backward fill*** (Cocok untuk data *time-series*).
   - ⊕ Menjaga kontinuitas waktu.
   - ⊖ Tak cocok untuk perubahan drastis.
4. **Teknik Interpolasi**.
   - ⊕ Menangkap tren/pola, mempertahankan hubungan antar data.

⊖ Mengasumsikan pola tertentu (*linear, kuadratik*), dan dapat menjadi kompleks.

5. **Imputasi Regresi**.

   ⊕ Lebih akurat dari *mean/median*, mempertahankan hubungan antar variabel.

   ⊖ Cenderung *overfitting*, mengurangi varians.

3 Statistika Deskriptif

Statistika Deskriptif adalah suatu teknik statistika untuk mengumpulkan, mengklasifikasikan, meringkas, dan menyajikan data agar mampu menggambarkan karakteristik data.

1. **Distribusi Frekuensi**

   Suatu cara untuk menunjukkan frekuensi atau banyak objek masing-masing kelas dan direpresentasikan dalam grafik atau tabel.

2. **Tendensi Sentral**

   Suatu teknik untuk merangkum dan mendeskripsikan kelompok variabel lewat nilai rata-rata dari kumpulan data. Tujuannya adakah untuk mengetahui kecenderungan letak pusat data.

   Distribusi simetris : Mean $\approx$ Median $\approx$ Modus
   Miring kanan : Mean $>$ Median $>$ Modus
   Miring kiri : Modus $>$ Median $>$ Mean

   Catatan : Jika mean jauh dari median, maka kemungkinan terdapat *outlier* dalam data.

3. **Variabilitas Data**

   Variabilitas berfungsi untuk dapat menganalisis persebaran distribusi dalam suatu kumpulan data.

   - **Range** : Jarak antara nilai terbesar dan terkecil
   - **Standar Deviasi** : Tingkat ketersebaran data, seberapa dekat data dengan mean
     $\sigma = \sigma^{\frac{1}{2}}$
   - **Varians** : Tingkat penyebaran data
     $\sigma = \frac{\sum(x-\mu)^2}{n}$

Contoh Implementasi pada Python

```
# Import libraries
import pandas as pd

df = pd.read_csv(r'datacontoh.csv')
df.head()

# Menentukan Mean Median dan Modus
```

```
mean_data = df['sepal_length'].mean()
median_data = df['sepal_length'].median()
modus_data = df['sepal_length'].mode()[0]

# Mencetak hasil
print(f'Mean: {mean_data}')
print(f'Median: {median_data}')
print(f'Modus: {modus_data}')

# Menentukan variabilitas data

# Jangkauan data pada kolom sepal_length
range_data = df['sepal_length'].max() - df['sepal_length'].min()

# Standard Deviasi pada kolom sepal_length
std_dev_data = df['sepal_length'].std()

# Variansi pada kolom sepal_length
variance_data = df['sepal_length'].var()

# Mencetak hasil
print(f'Jangkauan: {range_data}')
print(f'Standard Deviasi: {std_dev_data}')
print(f'Variansi: {variance_data}')
```

4 Visualisasi Data

Visualisasi data adalah sebuah proses merubah data mentah ke dalam bentuk visual, seperti grafik, diagram, peta, atau plot. **Tujuannya** adalah untuk memahami pola, tren, hubungan, atau *outlier*.

Library yang digunakan

- Matplotlib : Membuat grafik, line plot, *bar chart*, *scatter plot*, histogram
- Seaborn : Heatmap, pairplot, violinplot

Scatter Plot

- Mengamati hubungan antara 2 variabel
- Mengidentifikasi korelasi
- Menemukan *outlier*

Heatmap : Visualisasi data berbentuk peta warna yang merepresentasikan nilai dalam sebuah tabel atau matriks

- Menampilkan hubungan antarvariabel dalam matriks
- Analisis korelasi antarvariabel
- Memvisualisasikan data berdimensi banyak

Konsep heatmap Correlation

Correlation matrix adalah tabel yang berisi koefisien korelasi antar setiap pasangan variabel.

$$r_{XY} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2}\sqrt{\sum(y_i - \bar{y})^2}} \tag{1}$$

Untuk membentuk Heatmap Correlation, caranya adalah dengan mengubah rumus korelasi tersebut ke pemetaan warna $f : [-1, 1] \rightarrow$ warna.

Heatmap Correlation menggambarkan hubungan antarvariabel.

- 1 : Hubungan / korelasi yang kuat → Warna Gelap
- 0 : Tidak memiliki hubungan → Warna Netral
- -1 : Hubungan / korelasi berkebalikan yang kuat → Warna Cerah

Contoh Implementasi pada Python

```
# Import libraries
import pandas as pd
import matplotlib.pyplot as plt

# Membaca Data
df = pd.read_csv(datavisualisasi.csv)
df.head()

# Scatter Plot
plt.scatter(df['day'], df['tip'], c=df['size'],
            s=df['total_bill'])

# Menambahkan judul pada plot
plt.title('Scatter Plot of Day vs Tip')

# Mengatur label sumbu x dan y
plt.xlabel('Day')
plt.ylabel('Tip Amount')

plt.colorbar()

plt.show()

# Line Chart

# Scatter Plot
plt.plot(df['tip'])
plt.plot(df['size'])

# Kalo mau diubah menjadi bar, tinggal ubah plot menjadi bar
# Kalo mau diubah menjadi histogram, tinggal ubah plot menjadi hist

# Menambahkan judul pada plot
plt.title('Line Chart of Tip and Size')
```

```
# Mengatur label sumbu x dan y
plt.xlabel('Day')
plt.ylabel('Tip')

plt.show()

# Contoh menggunakan Seaborn
import seaborn as sns

sns.barplot(x='day',y='tip', data=df,
            hue='sex')

plt.show()

# Histoplot pada Seaborn
sns.histplot(x='total_bill', data=df, kde=True, hue='sex')

plt.show()

# Heatmap correlation matrix
co_mtx = df.corr(numeric_only=True)

# Mencetak correlation matrix
print(co_mtx)

# Plot correlation matrix menggunakan heatmap
sns.heatmap(co_mtx, annot=True, cmap="YlGnBu")
plt.title('Heatmap of Correlation Matrix')

# cmap bisa diganti sesuai keinginan, misal 'coolwarm', 'viridis',
   dll
plt.show()
```

5 Outlier Detection

Outlier adalah sebuah nilai dalam data yang sifatnya anomali karena sangat terpisah / jauh dari titik lain yang diobservasi dalam data. *Outlier* dapat muncul dikarenakan adanya kesalahan pencatatan, input, variasi alami, atau fenomena langka. Data yang dideteksi sebagai *outlier* memiliki kecenderungan untuk merusak mean, menurunkan performa model *machine learning*, menyebabkan kesalahan interpretasi. Akan tetapi, tidak semua *outlier* harus dibuang karena terkadang *outlier* tersebut **penting**, misal untuk *fraud detection*.

Terdapat beberapa jenis *outlier*, di antaranya adalah sebagai berikut.

- **Global Outliers** : Nilai jauh dari data lainnya
- **Contextual Outliers** : Nilai yang menyimpang dan memiliki makna tertentu
- **Collective Outliers** : Sekumpulan data yang menyimpang bersama

Deteksi Outliers

Outlier dalam data harus dideteksi karena penting untuk mengetahui makna keberadaan *outlier*, meningkatkan akurasi, menjaga kualitas data.

Metode untuk Deteksi Outliers

1. **Visualisasi**

   Melalui visualisasi data, *outlier* dapat diamati sebagai data yang "menyimpang" dan jauh dari data lainnya. Terdapat beberapa hal yang dapat digunakan untuk visualisasi data, di antaranya *box plot*, *scatter plot*, histogram, dan *density plot*.

2. **Z-score / Standar Deviasi**

   Teknik ini menghitung seberapa jauh data dari mean dengan menghitung standar deviasi. Biasanya data *outlier* digambarkan dalam $|Z| > 3$.

   $$Z = \frac{x - \mu}{\sigma}$$

   Cocok digunakan untuk data yang terdistribusi normal.

3. **Metode Interquartile Range (IQR)**

   Deteksi *outlier* dengan melihat sebaran 50% data tengah.

   $$\text{IQR} = \text{Q3} - \text{Q1}$$

   Suatu data dianggap sebagai *outlier* ketika tidak berada pada rentang:

   $$\text{Batas bawah} : \text{Q1} - 1,5 \times \text{IQR} \quad | \quad \text{Batas atas} : \text{Q3} + 1,5 \times \text{IQR}$$

6 Reduksi Dimensi dengan PCA

Reduksi dimensi adalah suatu teknik untuk mengurangi jumlah fitur dalam sebuah data untuk mempertahankan informasi yang paling relevan. Konsepnya adalah dengan mentransformasi fitur lama menjadi fitur baru (***Principal Component***) yang lebih ringkas.

Kenapa PCA Penting?

- **Overfitting** apabila terlalu banyak fitur
- **Waktu Komputasi Lama**
- **Interpretasi Sulit** ketika memiliki banyak dimensi
- **Redundansi Data** karena banyak fitur yang menyimpan informasi berulang

Langkah-Langkah dalam PCA

1. **Normalisasi Data**. Perbedaan skala antarfitur sangat memengaruhi hasil, sehingga perlu dinormalisasi.

2. **Membuat matriks kovarians**. Matriks untuk membantu menentukan bagaimana hubungan antarfitur saling berinteraksi.
3. **Menentukan Principal Components**. *Eigenvalue* & *eigenvector* terbesar akan menjadi **Principal Component** utama.
4. **Memilih Jumlah Dimensi Optimal**. Explained Variance Ratio digunakan untuk menentukan banyak variansi yang digunakan.
5. **Transformasi Data ke Dimensi Baru**. Mengubah *dataset* asli menjadi representasi baru.

Keterbatasan PCA

- Principal components sulit ditafsirkan maknanya
- Asumsi linear
- Sensitif terhadap skala fitur
- Tidak cocok jika fitur tak linear dan informasi ada pada varians kecil

7 Transformasi Data

Transformasi data adalah suatu proses untuk mengonversi dari data mentah ke data yang formatnya lebih sesuai untuk proses analisis.

Mengapa transformasi penting?

- Model lebih mudah belajar
- Performa meningkat
- Konvergensi lebih cepat
- Hasil lebih stabil dan akurat

Terdapat beberapa contoh transformasi data, di antaranya sebagai berikut.

1. **Normalisasi**

   Normalisasi bertujuan untuk menjaga proporsi nilai dalam suatu fitur. Normalisasi mengubah nilai menjadi dalam rentang tertentu $[0, 1]$. Proses ini biasanya digunakan untuk Neural Network, Visualisasi.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

2. **Standardisasi**

   Standardisasi mengubah suatu data agar memiliki mean = 0 dan std = 1. Transformasi ini biasanya digunakan untuk Linear / Logistic Regression, SVM, PCA, dan k-means.

$$z = \frac{x - \mu}{\sigma}$$

3. **Scaling**

   Scaling adalah proses normalisasi + standarisasi. Tujuannya adalah menyamakan skala antarfitur dan mencegah dominasi fitur bernilai besar. Scaling ini biasanya dapat digunakan pada Decision Tree, Random Forest, ataupun XGBoost.

8 Encoding

Encoding adalah proses mengubah data kategorikal (teks atau label) ke dalam bentuk numerik agar dapat dipahami oleh *machine learning* dan *data mining*.

Tujuan Encoding

- Menghindari kesalahan interpretasi model
- Mengubah data non-numerik $\rightarrow$ numerik
- Meningkatkan akurasi dan stabilitas model

Teknik-Teknik Data Encoding untuk Kategorikal

1. **Label Encoding**

   Mengubah masing-masing data kategorikal menjadi angka unik yang digunakan pada Decision Tree atau XGBoost.

2. **One-Hot Encoding**

   Mengonversi kategori ke dalam beberapa kolom biner dengan tiap kolom merepresentasikan kategori. Biasanya digunakan pada linear model, logistic regression, dan Neural Network

3. **Ordinal Encoding**

   Seperti label encoding, tetapi mempertahankan urutan yang cocok digunakan pada data dengan urutan.

4. **Target Encoding**

   Mengganti data kategori dengan rata-rata nilai target pada kategori tersebut.

5. **Binary Encoding**

   Mengonversi kategori menjadi representasi biner, kemudian dipisah menjadi beberapa kolom.

6. **Frequency Encoding**

   Mengubah setiap kategori dengan frekuensi kemunculannya di dalam *dataset*.

9 Menangani Imbalanced Data

Imbalanced Data adalah suatu kondisi ketika distribusi data pada tiap kelas tidak seimbang. Suatu kelas memiliki jumlah anggota yang tak seimbang dibandingkan kelas lainnya.

Kenapa harus diatasi?

Karena *imbalanced data* dapat mengakibatkan model memiliki performa yang buruk akibat adanya bias pada kelas mayoritas.

Cara untuk Mengatasi *Imbalanced Data*

1. **Menggunakan metode evaluasi yang sesuai**

   Accuracy kurang baik pada data tak seimbang karena dapat bernilai tinggi dengan hanya memprediksi kelas mayoritas. Untuk itu, metode yang lebih sesuai menggunakan metrik evaluasi Precision, Recall, dan F1-score.

2. **Resampling** : Upsampling dan Downsampling

   Mengubah jumlah sampel antarkelas supaya distribusi lebih seimbang. Caranya dengan cara menghapus beberapa data kelas mayoritas atau menambahkan data kelas minoritas secara acak.

3. **BalancedBagging Classifier**

   Mengatasi *imbalanced* data dengan menambahkan mekanisme penyeimbangan selama proses pelatihan.

4. **SMOTE** | Synthetic Minority Oversampling Technique

   SMOTE membuat data sintetis baru untk kelas minoritassehingga meningkatkan keberagaman data dengan menciptakan *instance* buatan.

   Konsep SMOTE : Mengamati data minoritas $\rightarrow$ memilih tetangga terdekat dengan k-*nearest neighbor* $\rightarrow$ *generate data*.

   Cara Generate : Hitung selisih vektor ($\Delta = x_{nn} - x_i$ lalu mengambil nilai acak (λ) dengan $\lambda \in (0, 1)$, kemudian membuat data baru ($x_{baru} = x_i + \lambda\Delta$)