Preprocessing Data

Tujuan Praktikum:

Pada akhir praktikum ini, Mahasiswa diharapkan dapat menerapkan praproses data dasar

Bahasan:

1. Mengetahui informasi dataset
2. Menganalisis tipe data
3. Pengecekan konsistensi data
4. Mengindentifikasi dan mengatasi missing value
5. Data standardization
6. Data normalization

Apakah tujuan dari praproses data?

Praproses data : proses transformasi atau pemetaan data dari satu bentuk asli atau mentah ke format lain yang lebih sesuai untuk dilakukan analisis lebih lanjut.

1. Informasi Dataset

Tahapan-tahapan dalam preprocessing dapat berbeda-beda, tergantung kepada permasalahan yang ada. Memahami informasi dari dataset menjadi salah satu bagian penting dalam praproses data.

Pada praktikum ini kita akan menggunakan dataset Titanic. Dataset ini berisi sejumlah daftar penumpang kapal Titanic yang menjadi korban dan selamat pada kecelakaan kapal tersebut.

Dataset ini digunakan untuk memprediksi apakah seseorang selamat atau tidak berdasarkan data-data lainnya. Adapun informasi fitur pada data ini dapat dilihat pada Tabel di bawah ini.

Variable	Definisi	Penjelasan Isi
survival	Kolom label yang menyatakan penumpang selamat atau tidak	0 = Tidak selamat, 1 = Selamat
pclass	Kelas tiket penumpang	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Jenis kelamin penumpang	
Age	Umur penumpang dalam tahun	
sibsp	Jumlah saudara kandung / pasangan di kapal Titanic	
parch	Jumlah orang tua / anak di kapal Titanic	
ticket	Nomor tiket	
fare	Tarif Penumpang	
cabin	Nomor Kabin	
embarked	Pelabuhan keberangkatan	C = Cherbourg, Q = Queenstown, S = Southampton

2. Menganalisis tipe data

Dataset ini terdiri atas dua dokumen yaitu train.csv dan test.csv yang dapat diunduh pada https://www.kaggle.com/competitions/titanic/overview.

Train.csv berisi data yang akan digunakan untuk pelatihan.

Test.csv adalah data yang akan digunakan untuk evaluasi.

Untuk memuat data ke dalam notebook kita dapat menggunakan pandas dengan fungsi read_csv dengan parameter nama file.

Pada kasus ini langkah tersebut dilakukan pada baris 11 dan 12. Setelah berhasil memuat data, kita tampilkan 5 data pertama dengan menggunakan fungsi head() (Baris 13).

Namun sebelum kita memulai melakukan analisis data, pastikan google colab anda sudah tersambung dengan google drive anda. Pastikan bahwa data yang akan digunakan pada praktikum ini, telah anda simpan di dalam google drive.

```
from google.colab import drive
import sys
import os
drive.mount('/content/gdrive')
drivePath = ('/content/gdrive/My Drive/Comvis/Pertemuan2')
sys.path.append(drivePath)
os.chdir(drivePath)
current_dir = os.getcwd()
current_dir
```

```
#data analysis dan wrangling
import pandas as pd
import numpy as np
import random as rnd

#visualization
import seaborn as sns
import matplotlib.pyplot as plt

train_df = pd.read_csv('train.csv')
test_df = pd.read_csv('test.csv')
train_df.head()
```

Berdasarkan 5 data pertama dari fungsi head, kita telah dapat menentukan tipe-tipe data. Adapun tipe data yang ada pada dataset ini adalah

- Kategori : Survived, Sex, dan Embarked.
- Ordinal : Pclass
- Continous: Age, Fare.
- Discrete: SibSp, Parch.

Tipe data Ticket merupakan gabungan antara numerik dan alpha-numerik Tipe data Cabin adalah alpha-numerik.

Ringkasan terkait tipe data

Tipe Data	Ciri-ciri	Contoh
Numerik	Hanya angka, dapat dihitung	25, 100.5, -3
Alfanumerik	Kombinasi huruf, angka, dan simbol	AB123, JKT-2024, X5Y9
Kategori	Data dalam bentuk kategori atau label tanpa urutan tertentu	Jenis Kelamin, Warna Mata
Ordinal	Data dengan urutan tetapi tanpa selisih yang pasti	Tingkat Kepuasan, Tingkat Pendidikan
Diskret	Bilangan bulat terbatas, tidak memiliki pecahan	Jumlah anak, Jumlah mobil
Kontinu	Data numerik yang bisa memiliki pecahan dalam rentang tertentu	Tinggi badan, Berat badan

3. Pengecekan konsistensi data

Berdasarkan analisa awal, kita memiliki tiga fitur yang bertipe kategori yaitu Survived, Sex, dan Embarked, dua discrete dan satu ordinal. Ketiga data tersebut akan dilihat sebaran nilainya dan konsistensi nilai (untuk kategori).

```
#pengaturan layout
n_rows = 3
n_cols = 2

#daftar kolom
kolom = ["Sex", "Embarked", "SibSp", "Parch", "Survived", "Pclass"]

fig, axes = plt.subplots(nrows=n_rows, ncols=n_cols, figsize=(13,10))
i=0
fig.suptitle('CountPlot Dataset Titanic', fontsize="28")
for row in range(n_rows):
  for col in range(n_cols):
    sns.countplot(x=kolom[i], data=train_df, ax=axes[row,col], hue=kolom[i], palette="bright", legend=False)

    #menampilkan label nilai per unique value
    for p, label in zip(axes[row,col].patches, train_df[kolom[i]].value_counts()):
      axes[row,col].annotate(label, (p.get_x(), p.get_height()))
    i=i+1
```

Pada fitur sex dan embark ini tidak ditemukan inkonsistensi data. Pada fitur Sex terdapat dua nilai unik yakni male dan female dan kolom Embarked memiliki tiga nilai unik yakni S,C dan Q. Contoh inkonsistensi data pada dataset adalah apabila misalnya pada sex ditemukan nilai seperti male, M, Female, FEMALE. Data yang bernilai male dan M adalah sama tetapi memiliki value yang berbeda. Solusinya adalah dengan menyatukan data-data yang sama menjadi nilai yang sama.

4. Mengindentifikasi dan mengatasi missing value

Ketika kita menampilkan 5 data pertama, kita mengetahui bahwa kolom Cabin memiliki data bernilai Nan.

Dalam komputasi, NaN adalah singkatan dari Not a Number. NaN adalah nilai tipe data numerik yang mewakili nilai yang tidak ditentukan atau tidak terwakili.

Nilai NaN ini salah satu tanda bahwa data kita memiliki missing value.

Lalu, bagaimana kita mengidentifikasi semua missing value dan mengatasinya? Bagaiaman bekerja dengan missing value?

Langkah-langkah :

- Mengidentifikasi missing data
- Menangani missing value
- Perbaiki format data

Sebelum kita mulai mengatasi missing value, kita perlu memilih kolom/fitur (seleksi fitur) mana saja yang akan digunakan untuk mencapai tujuan yang diharapkan. Pada data praktikum ini tujuan kita adalah memprediksi apakah seseorang selamat atau tidak berdasarkan data-data yang ada.

Pada dataset ini, kolom nama, tiket, Informasi kabin tidak digunakan sehingga kolom-kolom ini dapat dihapus

```
#Hapus kolom yang tidak digunakan
train_df = train_df.drop(["Name", "Ticket", "Cabin", "PassengerId"], axis=1)
train_df.head()
```

Selanjutnya, kita lakukan identifikasi kolom mana saja kah yang memiliki missing value.

Missing value dikonversi ke nilai default python, kemudian kita gunakan fungsi built in python untuk mengidentifikasi.

Terdapat 2 metode untuk **mendeteksi missing value**:

1. **.isnull()**
2. **.notnull()**

```
train_df.info()
```

```
train_df.isnull().sum()
```

Dari ringkasan di atas, diketahui bahwa kolom Age memiliki 177 missing value dan kolom Embarked memiliki 2 missing value.

Bagaimana menangani missing value?

1. drop data

- drop seluruh baris
- drop seluruh kolom

2. replace data

- replace dengan mean
- replace dengan frequency
- replace dengan fungsi lain

Seluruh kolom di drop hanya jika hampir seluruh isi kolom tersebut missing. Dalam dataset kita tidak ada kolom dengan karakteristik tersebut.

Kolom Age memiliki 177 missing value yang bisa kita tangani dengan menggantikan nilai missing dengan nilai median.

Kolom Embarked memiliki 2 missing value yang bisa kita tangani dengan melakukan drop baris.

```
train_df['Age'] = train_df['Age'].fillna(train_df['Age'].median())
```

```
train_df.info()
```

```
train_df.isnull().sum()
```

```
train_df.head()
```

```
train_df.tail()
```

```
#drop baris dengan Nan dalam kolom Embarked
train_df.dropna(subset=["Embarked"], axis=0, inplace=True)

#reset index, karena kita akan drop 2 baris
train_df.reset_index(drop=True, inplace=True)
```

```
train_df.info()
train_df.isnull().sum()
train_df.tail()
```

5. Data Standardization

Data biasanya diperoleh dari berbagai sumber dan berbagai format. (hati-hati: data standardization juga merupakan istilah dari normalisasi data dengan dikurangi mean dan dibagi dengan standar deviasi).

Apa itu standarisasi?

Standarisasi adalah proses transformasi data ke dalam format yang mmebuat peneliti dapat membuat perbandingan yang berguna/bermakna.

Contoh: Misal data cuaca yang anda miliki dalam satuan Fahrenheit (F), sedangkan umumnya suhu itu umumnya menggunakan satuan Celcius (C). Maka kita perlu melakukan data transformation untuk mentransformasi Fahrenheit menjadi Celcius.

Pada dataset praktikum kali ini kita tidak menemukan kondisi seperti ini.

6. Data Encoding

Data encoding adalah proses mengubah data ke dalam format tertentu yang dapat lebih mudah diproses oleh model atau sistem komputer. Dalam konteks analisis data dan machine learning, encoding sering kali digunakan untuk mengonversi data kategorikal (seperti teks atau label) menjadi format numerik yang dapat digunakan oleh algoritma untuk analisis lebih lanjut atau prediksi.

Ada beberapa jenis teknik data encoding yang umum digunakan, terutama untuk data kategorikal, di antaranya:

1. Label Encoding.

Label encoding adalah metode yang mengonversi kategori menjadi angka (numerik). Setiap kategori unik diberi angka mulai dari 0 hingga n-1, di mana n adalah jumlah kategori dalam fitur.

Contoh: Misalkan kita memiliki kolom "warna" dengan nilai kategorikal seperti:

```
['merah', 'biru', 'hijau', 'merah', 'biru']
```

Setelah label encoding, nilai-nilai tersebut akan diubah menjadi:

```
[0, 1, 2, 0, 1]
```

Di sini:

'merah' menjadi 0 'biru' menjadi 1 'hijau' menjadi 2

2. One-Hot Encoding

One-hot encoding adalah metode yang mengonversi setiap kategori menjadi sebuah kolom biner (0 atau 1), di mana setiap kategori menjadi kolom tersendiri. Untuk setiap baris, kolom yang sesuai dengan kategori yang ada akan diberi nilai 1, sedangkan kolom lainnya diberi nilai 0.

Misalkan kita memiliki kolom "warna" dengan nilai kategorikal seperti:

```
['merah', 'biru', 'hijau', 'merah', 'biru']
```

Setelah one-hot encoding, kita akan mendapatkan:

```
merah menjadi [1 0 0]
biru menjadi [0 1 0]
hijau menjadi [0 0 1]
```

Pada dataset ini, kita perlu melakukan data encoding yakni mengubah data kategori menjadi data numerik.

Kita dapat menggunakan fungsi map pada dataframe. Pada kolom sex, female akan diubah menjadi 0 dan male menjadi 1. Begitu juga untuk Embarked S=0, C=1 dan Q=2

```
train_df.Sex = train_df.Sex.map({'female':0, 'male':1})
train_df.Embarked = train_df.Embarked.map({'S':0, 'C':1, 'Q':2})
train_df.head()
```

7. Data Normalization

Mengapa perlu normalisasi?

Normalisasi merupakan proses mentransformasi nilai dari beberapa variabel ke dalam range yang sama. Normalisasi melakukan scaling pada variabel rata-ratanya 0, variance nya 0, atau menskalakan variabel supaya berada dalam range 0-1. Pada dataset ini kita perlu melakukan normalisasi pada kolom Age dan Fare.

Teknik normalisasi ini digunakan dalam machine learning untuk meningkatkan kinerja model dengan mengurangi perbedaan skala antar fitur.

Teknik normalisasi yang sering digunakan:

1. Simple Feature Scaling

2. Min-Max Scalling (Min-Max Normalization)
3. Z-Score Normalization

```
#Simple Feature Scaling
df_simplescaler = train_df.copy()

df_simplescaler['Age'] = df_simplescaler['Age']/df_simplescaler['Age'].max()
df_simplescaler['Fare'] = df_simplescaler['Fare']/df_simplescaler['Fare'].max()

df_simplescaler.head()


#Min-Max Scalling
df_minmax1 = train_df.copy()
df_minmax1['Age'] = (df_minmax1['Age']-df_minmax1['Age'].min())/(df_minmax1['Age'].max()-df_minmax1['Age'].min())
df_minmax1['Fare'] = (df_minmax1['Fare']-df_minmax1['Fare'].min())/(df_minmax1['Fare'].max()-df_minmax1['Fare'].min())

df_minmax1.head()


#atau anda bisa menggunakan sklearn untuk melakukan normalisasi
from sklearn.preprocessing import MinMaxScaler

df_minmax2 = train_df.copy()
scaler = MinMaxScaler()
df_minmax2[['Age', 'Fare']]= scaler.fit_transform(df_minmax2[['Age', 'Fare']])


df_minmax2.head()


#Z-Score Scalling
df_zscore1 = train_df.copy()
df_zscore1['Age'] = (df_zscore1['Age'] - df_zscore1['Age'].mean()) / df_zscore1['Age'].std(ddof=0) #ddof = 0 berarti menggunakan standar
df_zscore1['Fare'] = (df_zscore1['Fare'] - df_zscore1['Fare'].mean()) / df_zscore1['Fare'].std(ddof=0)

df_zscore1.head()


#atau anda bisa menggunakan sklearn untuk melakukan normalisasi
from sklearn.preprocessing import StandardScaler

df_zscore2 = train_df.copy()

scaler = StandardScaler()
df_zscore2[['Age', 'Fare']] = scaler.fit_transform(df_zscore2[['Age', 'Fare']])

df_zscore2.head()
```

TUGAS

Lakukan praproses data pada studi kasus: Pengumpulan data dengan metode scrapping berita detik.com yang terdapat pada modul p2_preprocessing data