



BY DAFFA ANDIKA ZAIN - 151

Topic 5 & 6 - Assignment

Hands-On Python 2

P Y T H O N F O R D A T A A N A L Y S I S

Data Cleansing and Data Preprocessing

Data cleansing adalah proses memodifikasi atau menghapus data yang dianggap tidak akurat, duplikat, tidak lengkap, salah format, maupun rusak dalam kumpulan data yang dimiliki.

Data Preprocessing adalah teknik yang digunakan untuk mengubah data mentah dalam format yang berguna dan efisien.

Dengan begitu data berkualitas, bisnismu bisa memperoleh insights yang jitu. Pengaplikasian dari insights tersebut akan mencegah munculnya rasa frustrasi pelanggan dan karyawan, berpotensi meningkatkan produktivitas, hingga bisa membantu meningkatkan kualitas analisis data dalam pengambilan keputusan.





Telco Churn Dataset

Content

Each row represents a customer, each column contains customer's attributes described on the column Metadata.

The data set includes information about:

- Customers who left within the last month – the column is called Churn
- Services that each customer has signed up for – phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies
- Customer account information – how long they've been a customer, contract, payment method, paperless billing, monthly charges, and total charges
- Demographic info about customers – gender, age range, and if they have partners and dependents

Topic 5 assignment:

Lakukan beberapa langkah berikut pada dataset telco churn

- Missing Values Checking
- Categorical Data Encoding
- Anomalies and Outlier Handling

Importing library dan mount drive

```
▶ #biasanya nnti ada kode yang abis dirun ada warningnya maka kita bisa menghilangkannya dengan kode berikut
import warnings
warnings.filterwarnings('ignore')

#import library python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

[ ] from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

Read dataset berformat csv ke dalam dataframe dengan pandas

```
[ ] df = pd.read_csv("/content/drive/MyDrive/PZSIB - DATA ANALYTICS/Telco-Customer-Churn.csv", encoding='utf-8')
df.head()
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	... Contract	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	CreditCard	PaperlessBilling	MonthlyCharges	TotalCharges
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	... Contract	No	No	No	No	No	No	M	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	... Contract	Yes	No	No	No	No	No	C	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	... Contract	No	No	No	No	No	No	M	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	... Contract	Yes	Yes	No	No	No	No	C	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	... Contract	No	No	No	No	No	No	M	

5 rows × 21 columns

1. Missing Values Checking

Note: Mengecek pada dataset tersebut apakah masih ada data yang missing dan menghandle missing value tersebut agar data yang digunakan dapat meningkatkan efisiensi kerja karena proses ini akan memudahkan Anda dan tim pengolah data untuk menemukan apa yang dibutuhkan dari data

Langkah 1: melihat informasi dataframe

Code:

```
▶ #melihat informasi pada dataframe  
df.info()
```

Penjelasan:

.info() merupakan fungsi dari library pandas yang digunakan untuk melihat informasi yang ada pada dataframe.

dapat kita lihat bahwa:

1. Dataset memiliki 21 kolom dan 7043 baris,
2. TotalCharge memiliki type data yang tidak sesuai yaitu object yang seharusnya float maka kita harus menyesuaikan typedata dengan isi dari data terlebih dulu.

Output:

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 7043 entries, 0 to 7042  
Data columns (total 21 columns):  
 #   Column           Non-Null Count  Dtype     
---  --  
 0   customerID      7043 non-null   object    
 1   gender          7043 non-null   object    
 2   SeniorCitizen   7043 non-null   int64     
 3   Partner         7043 non-null   object    
 4   Dependents     7043 non-null   object    
 5   tenure          7043 non-null   int64     
 6   PhoneService    7043 non-null   object    
 7   MultipleLines   7043 non-null   object    
 8   InternetService 7043 non-null   object    
 9   OnlineSecurity  7043 non-null   object    
 10  OnlineBackup    7043 non-null   object    
 11  DeviceProtection 7043 non-null   object    
 12  TechSupport    7043 non-null   object    
 13  StreamingTV     7043 non-null   object    
 14  StreamingMovies  7043 non-null   object    
 15  Contract        7043 non-null   object    
 16  PaperlessBilling 7043 non-null   object    
 17  PaymentMethod   7043 non-null   object    
 18  MonthlyCharges  7043 non-null   float64   
 19  TotalCharges    7043 non-null   object    
 20  Churn           7043 non-null   object    
dtypes: float64(1), int64(2), object(18)  
memory usage: 1.1+ MB
```

Langkah 2: Mengubah type data

Code:

```
df["TotalCharges"] = pd.to_numeric(df["TotalCharges"], errors="coerce")
df.dtypes
```

Penjelasan:

Karena value pada kolom atau field TotalCharges belum sesuai maka kita harus sesuaikan terlebih dahulu

pd.to_numeric merupakan library pandas yang digunakan untuk mengonversi argumen yang diteruskan ke tipe numerik. Jenis pengembalian default dari fungsi tersebut adalah float64 atau int64 tergantung pada inputnya. Anda dapat menggunakan parameter downcast jika ingin mengonversi data ke tipe tertentu.

errors ="coerce" berfungsi untuk mengubah secara otomatis menjadi float/int sesuai dengan isi data atau isi value tersebut

lalu kita tampilkan kembali untuk melihat tipedatanya dengan .dtype. dapat kita lihat bahwa:

1. TotalCharge memiliki type data yang sudah sesuai yaitu dari object/string menjadi float.

Output:

customerID	object
gender	object
SeniorCitizen	int64
Partner	object
Dependents	object
tenure	int64
PhoneService	object
MultipleLines	object
InternetService	object
OnlineSecurity	object
OnlineBackup	object
DeviceProtection	object
TechSupport	object
StreamingTV	object
StreamingMovies	object
Contract	object
PaperlessBilling	object
PaymentMethod	object
MonthlyCharges	float64
TotalCharges	float64
Churn	object
dtype:	object

Langkah 3 : melihat value null pada dataframe (sebelum)

Code:

```
▶ #mengecek value null pada dataset dengan menggunakan .isnull() dan .sum()
df.isnull().sum()
```

Penjelasan:

.isnull() merupakan fungsi dari library pandas yang digunakan untuk membantu ditaris mana dan dikolom mana yang terdapat missing values pada dataframe.

.sum() merupakan fungsi aggregat untuk menjumlahkan values

maka, kita kombinasikan kedua fungsi tersebut untuk menampilkan total jumlah missing value yang terdapat pada dataframe

dapat kita lihat bahwa:

1. Sebelum tipe data TotalCharges dirubah, semuanya berjumlah 0 artinya Tidak ada data yang missing pada dataframe tersebut

Output:

```
customerID      0
gender          0
SeniorCitizen   0
Partner         0
Dependents     0
tenure          0
PhoneService    0
MultipleLines   0
InternetService 0
OnlineSecurity  0
OnlineBackup    0
DeviceProtection 0
TechSupport     0
StreamingTV    0
StreamingMovies 0
Contract        0
PaperlessBilling 0
PaymentMethod   0
MonthlyCharges  0
TotalCharges    0
Churn           0
dtype: int64
```

Langkah 4 : melihat value null pada dataframe (setelah)

Code:

```
▶ #mengecek value null pada dataset dengan menggunakan .isnull() dan .sum()
df.isnull().sum()
```

Penjelasan:

.isnull() merupakan fungsi dari library pandas yang digunakan untuk membantu diberi tahu di baris mana dan di kolom mana yang terdapat missing values pada dataframe.

.sum() merupakan fungsi aggregat untuk menjumlahkan values

maka, kita kombinasikan kedua fungsi tersebut untuk menampilkan total jumlah missing value yang terdapat pada dataframe

dapat kita lihat bahwa:

1. Setelah tipe data TotalCharges dirubah, TotalCharges berjumlah 11 artinya ada data yang missing pada dataframe tersebut

Output:

```
customerID      0
gender          0
SeniorCitizen   0
Partner         0
Dependents      0
tenure          0
PhoneService    0
MultipleLines   0
InternetService 0
OnlineSecurity   0
OnlineBackup     0
DeviceProtection 0
TechSupport      0
StreamingTV     0
StreamingMovies  0
Contract        0
PaperlessBilling 0
PaymentMethod    0
MonthlyCharges   0
TotalCharges     11
Churn           0
dtype: int64
```

Penjelasan tambahan:

Penjelasan:

ini membuktikan bahwa kesalahan semacam ini dikarenakan ada missing values pada kolom price tersebut. Blank cell atau karakter lainnya yang merepresentasikan missing values dianggap object (string) sehingga secara keseluruhan kolom tersebut juga akan dianggap bertipe data object, karena ada string dan integer. Jadi, ketika kita ingin melihat missing value pada suatu dataframe tadi missing value atau blank tidak dapat didetect dengan hanya menggunakan kode tersebut.

maka Pengecekan tipe data untuk tiap features (kolom) merupakan hal yang sangat penting untuk analisis dan eksplorasi data ke depannya. Maka ketika menemukan ketidaksesuaian tipe data seperti kasus di atas, kita harus mengkonversinya menjadi tipe data yang seharusnya.

Selanjutnya melakukan penghapusan atau drop data pada value yang blank atau missing value. Ada 3 cara untuk menghandle missing value pada suatu dataframe yaitu:

1. Menghapus atau drop
2. Replace
3. Membiarkannya

pada kasus ini kita dapat melakukan penghapusan atau drop karena data tersebut kecil dan tidak terlalu berpengaruh atau impactnya tidak terlalu besar dan data tersebut tetap akan berguna.

Langkah 5 : handling missing values dengan drop

Code:

```
[ ] # # Hapus baris dari fitur yang memiliki nilai yang hilang  
df.dropna(subset=['TotalCharges'], inplace=True)
```

Penjelasan:

.dropna merupakan sebuah metode atau fungsi yang digunakan untuk menghapus row yang berisi Null values atau missing values.

subset digunakan untuk mengakses elemen atau kolom mana yang ingin dibuat perubahan.

inplace=true adalah cara merubah dataframe tersebut langsung pada dataframanya tanpa harus mendefine dataframe lain terlebih dahulu.

disini kita menghapus row yang memiliki value kosong yang terdapat pada kolom TotalCharges dan bukan hanya menghapus dikolom tersebut tapi satu kesatuan barisnya yang terdapat missing value diTotalCharges tersebut.

note : "When inplace = True , the data is modified in place, which means it will return nothing and the dataframe is now updated. When inplace = False , which is the default, then the operation is performed and it returns a copy of the object. You then need to save it to something"

Langkah 6 : melihat value null kembali pada dataframe

Code:

```
[17] #mengecek value null kembali pada dataset dengan menggunakan .isnull() dan .sum()  
df.isnull().sum()
```

Penjelasan:

.isnull() merupakan fungsi dari library pandas yang digunakan untuk membantu dbaris mana dan dikolom mana yang terdapat missing values pada dataframe.

.sum() merupakan fungsi aggregat untuk menjumlahkan values

maka, kita kombinasikan kedua fungsi tersebut untuk menampilkan total jumlah missing value yang terdapat pada dataframe

dapat kita lihat bahwa:

- Dalam DataFrame sudah tidak terdapat value kosong yang artinya penghapusan pada dataframe sudah berhasil dan handling missing value juga sudah sukses.

Output:

```
customerID      0  
gender          0  
SeniorCitizen   0  
Partner          0  
Dependents      0  
tenure           0  
PhoneService     0  
MultipleLines    0  
InternetService  0  
OnlineSecurity   0  
OnlineBackup      0  
DeviceProtection 0  
TechSupport       0  
StreamingTV      0  
StreamingMovies   0  
Contract          0  
PaperlessBilling 0  
PaymentMethod     0  
MonthlyCharges    0  
TotalCharges      0  
Churn             0  
dtype: int64
```

Langkah 7 : melihat shape dari dataframe yang telah dilakukan modify

Code:

```
#melihat shape pada dataframe setelah didrop rownya  
#dari 7043 -> 7032  
df.shape
```

Output:

```
(7032, 21)
```

Penjelasan:

.shape berfungsi untuk melihat dimensi objek atau dataframe

dapat kita lihat bahwa:

- Pada dataframe terlihat bahwa barisnya atau rownya telah berkurang dari 7043 menjadi 7032 hal ini disebabkan karena, kita sudah drop 11 row yang memiliki value kosong pada kolom TotalCharges.
- dan total kolomnya masih 21 kolom

2. Categorical Data Encoding

Note:

Categorical Encoding adalah proses di mana kami mengubah data kategorikal menjadi data numerik. Kita semua tahu bahwa mesin tidak dapat memahami data kategorikal. Mesin membutuhkan semua variabel independen dan dependen yaitu fitur input dan output menjadi numerik. Ini berarti bahwa jika data kita mengandung variabel kategori, kita harus menyandikannya ke angka sebelum kita memasukkan data kita ke model. Karena sebagian besar model pembelajaran mesin hanya menerima variabel numerik, pemrosesan awal variabel kategori menjadi langkah yang diperlukan. Kita perlu mengonversi variabel kategori ini menjadi angka sedemikian rupa sehingga model dapat memahami dan mengekstrak informasi yang berharga.

"simplenya kita bakal transform data categorical jadi numerik agar lebih mudah untuk dilakukan pemrosesan dan analisis pada data tersebut".

terdapat 2 macam jenis data dalam categorical data encoding:

1. Data ordinal = data yang memiliki tingkatan atau urutan, contoh categorical data encoding dengan Label encoding dan Ordinal encoding
2. Data nominal = data yang tidak memiliki tingkatan atau urutan, contoh categorical data encoding dengan One Hot encoding

Langkah 1: Tampilkan dulu 15 data untuk melihat isi dari data pada dataframe

Code:

```
[ ] df.head(15)
```

Penjelasan:

Kita lihat terlebih dahulu data mana yang akan kita lakukan encoding sesuai dengan jenis datanya pada data frame yang telah kita lakukan missing value checking dan handling sebelumnya atau data yang sudah bersih dari missing values.

.head digunakan untuk menampilkan data top pada dataframe

Output:

	customerID	gender	SeniorCitizen	Partner	Dependents	Tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	No	No	No	One year	No	Mailed check	56.95	1889.50	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	Yes
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	Yes	No	No	One year	No	Bank transfer (automatic)	42.30	1840.75	No
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	No	No	No	Month-to-month	Yes	Electronic check	70.70	151.65	Yes
5	9305-CDSKC	Female	0	No	No	8	Yes	Yes	Fiber optic	No	...	Yes	No	Yes	Yes	Month-to-month	Yes	Electronic check	99.65	820.50	Yes
6	1452-KIOVK	Male	0	No	Yes	22	Yes	Yes	Fiber optic	No	...	No	No	Yes	No	Month-to-month	Yes	Credit card (automatic)	89.10	1949.40	No
7	6713-OKOMC	Female	0	No	No	10	No	No phone service	DSL	Yes	...	No	No	No	No	Month-to-month	No	Mailed check	29.75	301.90	No
8	7892-POOKP	Female	0	Yes	No	28	Yes	Yes	Fiber optic	No	...	Yes	Yes	Yes	Yes	Month-to-month	Yes	Electronic check	104.80	3046.05	Yes
9	6388-TABGU	Male	0	No	Yes	62	Yes	No	DSL	Yes	...	No	No	No	No	One year	No	Bank transfer (automatic)	56.15	3487.95	No
10	9763-GRSKD	Male	0	Yes	Yes	13	Yes	No	DSL	Yes	...	No	No	No	No	Month-to-month	Yes	Mailed check	49.95	587.45	No
11	7469-LKBCI	Male	0	No	No	16	Yes	No	No	No internet service	...	No internet service	No internet service	No internet service	No internet service	Two year	No	Credit card (automatic)	18.95	326.80	No
12	8091-TTVAX	Male	0	Yes	No	58	Yes	Yes	Fiber optic	No	...	Yes	No	Yes	Yes	One year	No	Credit card (automatic)	100.35	5681.10	No
13	0280-XJGEX	Male	0	No	No	49	Yes	Yes	Fiber optic	No	...	Yes	No	Yes	Yes	Month-to-month	Yes	Bank transfer (automatic)	103.70	5036.30	Yes
14	5129-JLPIS	Male	0	No	No	25	Yes	No	Fiber optic	Yes	...	Yes	Yes	Yes	Yes	Month-to-month	Yes	Electronic check	105.50	2686.05	No



Langkah 2: Lakukan Categorical data encoding pada data berjenis ordinal

Code:

```
df['Partner'] = df['Partner'].astype('category').cat.codes
df['Dependents'] = df['Dependents'].astype('category').cat.codes
df['PhoneService'] = df['PhoneService'].astype('category').cat.codes
df['PaperlessBilling'] = df['PaperlessBilling'].astype('category').cat.codes
df['Churn'] = df['Churn'].astype('category').cat.codes
df['InternetService'] = df['InternetService'].astype('category').cat.codes
df['MultipleLines'] = df['MultipleLines'].astype('category').cat.codes
df['OnlineSecurity'] = df['OnlineSecurity'].astype('category').cat.codes
df['OnlineBackup'] = df['OnlineBackup'].astype('category').cat.codes
df['DeviceProtection'] = df['DeviceProtection'].astype('category').cat.codes
df['TechSupport'] = df['TechSupport'].astype('category').cat.codes
df['StreamingTV'] = df['StreamingTV'].astype('category').cat.codes
df['StreamingMovies'] = df['StreamingMovies'].astype('category').cat.codes
df['Contract'] = df['Contract'].astype('category').cat.codes
df.head(3)
```

Output:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	... Contract	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn	
0	7590-VHVEG	Female	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	Electronic check	29.85	29.85	0
1	5575-GNVDE	Male	0	0	0	34	1	0	0	2	0	2	0	0	0	0	0	0	Mailed check	56.95	1889.50	0
2	3668-QPYBK	Male	0	0	0	2	1	0	0	2	0	0	0	0	0	0	0	1	Mailed check	53.85	108.15	1

Penjelasan:

Untuk Categorical data encoding pada data berjenis ordinal dapat menggunakan label encoding atau ordinal encoding namun pada case ini kita memakai label encoding pada data frame yang telah kita lakukan missing value checking dan handling sebelumnya atau data yang sudah bersih dari missing values.

#Label encoding mengubah setiap nilai dalam kolom menjadi angka yang berurutan agar lebih mudah diproses dan label harus merepresentasikan data awal atau sebelum diubah

#Label encoding juga memudahkan komputer dalam membaca sebuah data karena data-data yang dapat dibaca adalah data-data numerik maka kita harus mengubah data categorical -> numerical

#This type of encoding is used when the variables in the data are ordinal, ordinal encoding converts each label into integer values and the encoded data represents the sequence of labels.

note: makin banyak kolom makin ngga bagus buat proses datanya buat one hot encoding karena bakal banyak data yang 0 atau 1

Langkah 3 : Lakukan Categorical data encoding pada data berjenis nominal "PaymentMethod"

Code:

```
dummies_PaymentMethod = pd.get_dummies(df['PaymentMethod'],prefix='PaymentMethod')  
dummies_PaymentMethod.head()
```

Output:

	PaymentMethod_Bank transfer (automatic)	PaymentMethod_Credit card (automatic)	PaymentMethod_Electronic check	PaymentMethod_Mailed check	
0	0	0	1	0	0
1	0	0	0	0	1
2	0	0	0	0	1
3	1	0	0	0	0
4	0	0	0	1	0

Penjelasan:

untuk jenis Categorical Data Encoding yang tidak dapat menggunakan cara "Label Encoding" maka, dapat menggunakan "One hot encoding" jika data tersebut memiliki category lebih dari 2 yakni Paymentmethod dan gender.

```
#kondisi ini memudahkan algoritma machine learning membagi category dengan numerik  
#This type of encoding is used when the data is nominal  
.get_dummies berfungsi untuk membuat data dummy dengan metode yang kita ingin masukan
```

note: makin banyak kolom makin ngga bagus buat proses datanya buat one hot encoding karena bakal banyak data yang 0 atau 1

Langkah 4 : Menggabungkan data dummy kedalam dataframe utama

Code:

```
df = pd.concat([df, dummies_PaymentMethod], axis=1)  
df.head()
```

Penjelasan:

.concat berfungsi untuk menggabungkan dataframe dengan dataframe dummy sehingga tidak perlu dibedakan

Output:

PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn	PaymentMethod_Bank transfer (automatic)	PaymentMethod_Credit card (automatic)	PaymentMethod_Electronic check	PaymentMethod_Mailed check
1	Electronic check	29.85	29.85	0	0	0	1	0
0	Mailed check	56.95	1889.50	0	0	0	0	1
1	Mailed check	53.85	108.15	1	0	0	0	1
0	Bank transfer (automatic)	42.30	1840.75	0	1	0	0	0
1	Electronic check	70.70	151.65	1	0	0	1	0

Langkah 5 : Hapus kolom Payment method

Code:

```
df = df.drop('PaymentMethod',axis=1)  
df.head()
```

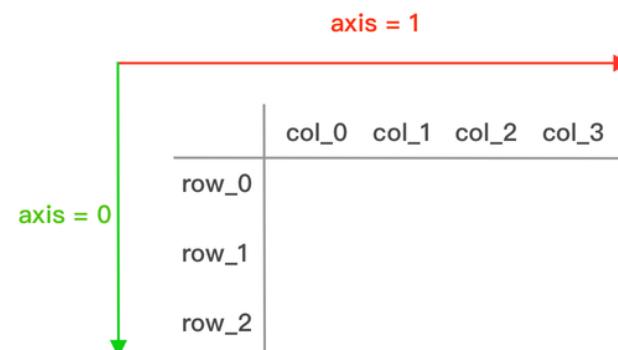
Penjelasan:
menghapus payment method karena sudah dibuat one hot encoding untuk membagi category PaymentMethod tersebut dengan fungsi .drop()

.drop() dapat digunakan untuk menghapus kolom pada dataframe dengan

note: penggunaan axis =1 itu ke kolom kalau axis=0 atau default itu menghapus index

Output:

PaperlessBilling	MonthlyCharges	TotalCharges	Churn	PaymentMethod_Bank transfer (automatic)	PaymentMethod_Credit card (automatic)	PaymentMethod_Electronic check	PaymentMethod_Mailed check
1	29.85	29.85	0	0	0	1	0
0	56.95	1889.50	0	0	0	0	1
1	53.85	108.15	1	0	0	0	1
0	42.30	1840.75	0	1	0	0	0
1	70.70	151.65	1	0	0	1	0



Langkah 6 : Lakukan Categorical data encoding pada data berjenis nominal "PaymentMethod"

Code:

```
dummies_gender = pd.get_dummies(df['gender'],prefix='gender')  
dummies_gender.head()
```

Penjelasan:

encoding kolom gender (sama seperti PaymentMethod)

Note: agar nama column nya sesuai maka isi row atau baris dikolom tersebut jangan diubah ke encoding lain maka terlebih dahulu agar encoding dengan one hot akan membaginya kedalam kategori-kategori dengan nilai binary dan dengan cara encoding sesuai jenis data

Output:

	gender_Female	gender_Male
0	1	0
1	0	1
2	0	1
3	0	1
4	1	0

Langkah 7 : Menggabungkan data dummy kedalam dataframe utama

Code:

```
df = pd.concat([df, dummies_gender], axis=1)  
df.head()
```

Penjelasan:

.concat berfungsi untuk menggabungkan dataframe dengan dataframe dummy sehingga tidak perlu dibedakan

Output:

PaymentMethod_Mailed check	gender_Female	gender_Male
0	1	0
1	0	1
1	0	1
0	0	1
0	1	0

Langkah 8 : Hapus kolom gender

Code:

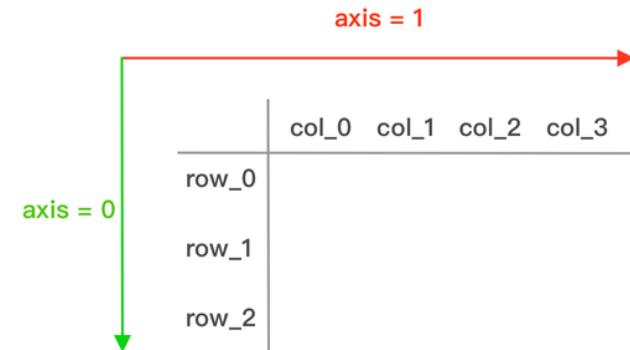
```
df = df.drop('gender',axis=1)  
df.head()
```

Penjelasan:

menghapus payment method karena sudah dibuat one hot encoding untuk membagi category PaymentMethod tersebut dengan fungsi .drop()

.drop() dapat digunakan untuk menghapus kolom pada dataframe dengan

note: penggunaan axis =1 itu ke kolom kalau axis=0 atau default itu menghapus index



Output:

PaymentMethod_Mailed check	gender_Female	gender_Male
0	1	0
1	0	1
1	0	1
0	0	1
0	1	0

3. Anomalies and Outlier Handling

Langkah 1: Untuk mengecek adanya perbedaan gap sebuah value/karakteristik

Code:

```
df.describe(['tenure', 'MonthlyCharges', 'TotalCharges'])
```

Penjelasan:

Untuk mengecek adanya perbedaan gap sebuah value/karakteristik pada dataset, data yang dipilih adalah data yang tidak dimanipulasi menjadi numerik.

Menggunakan rumus df.describe() = berfungsi untuk menampilkan data yang dipanggil

Output:

	tenure	MonthlyCharges	TotalCharges
count	7032.000000	7032.000000	7032.000000
mean	32.421786	64.798208	2283.300441
std	24.545260	30.085974	2266.771362
min	1.000000	18.250000	18.800000
25%	9.000000	35.587500	401.450000
50%	29.000000	70.350000	1397.475000
75%	55.000000	89.862500	3794.737500
max	72.000000	118.750000	8684.800000

Langkah 2: Mencari anomali handling melalui distribusi grafik histogram

Code:

```
f,ax = plt.subplots(figsize=(10,5))
sns.distplot(df['tenure'])

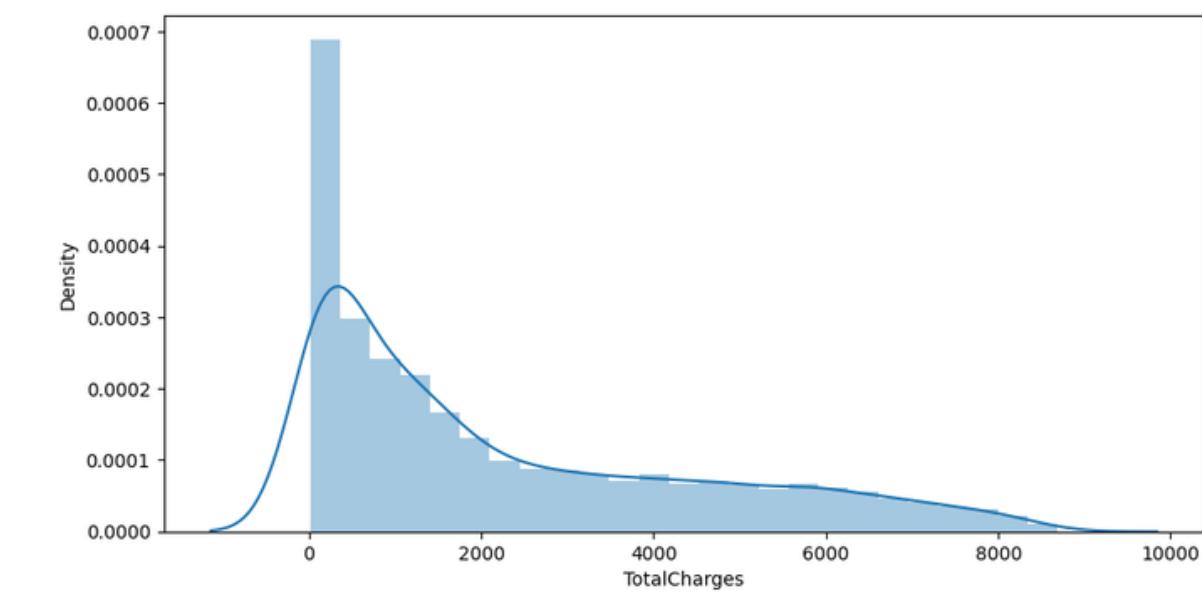
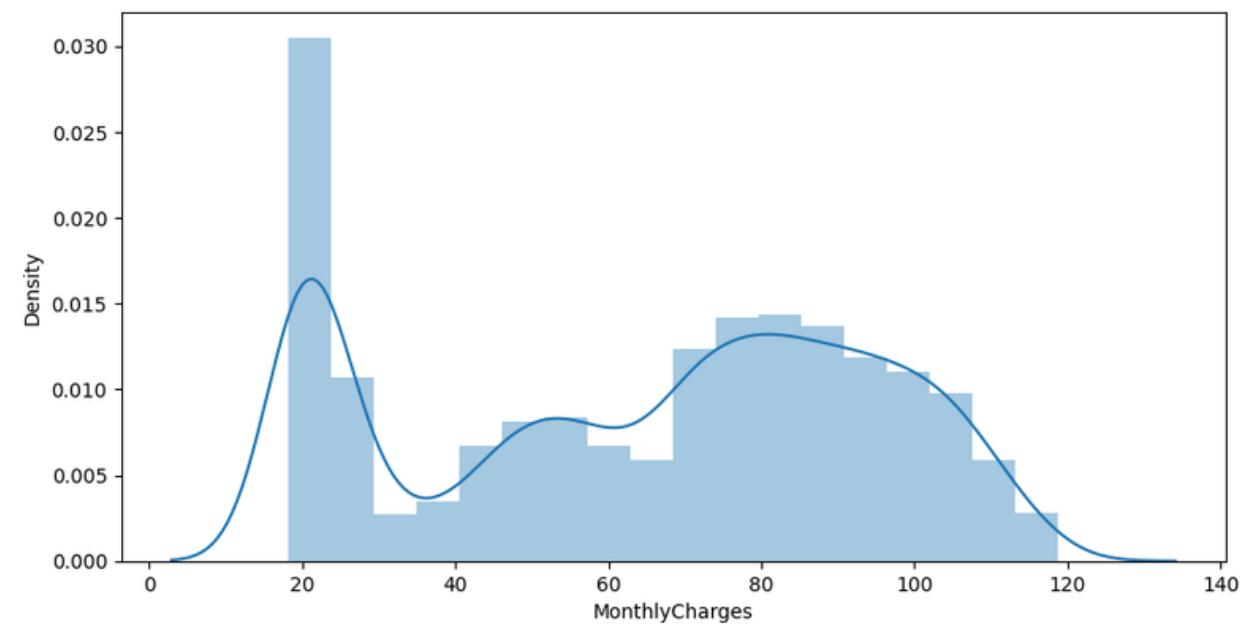
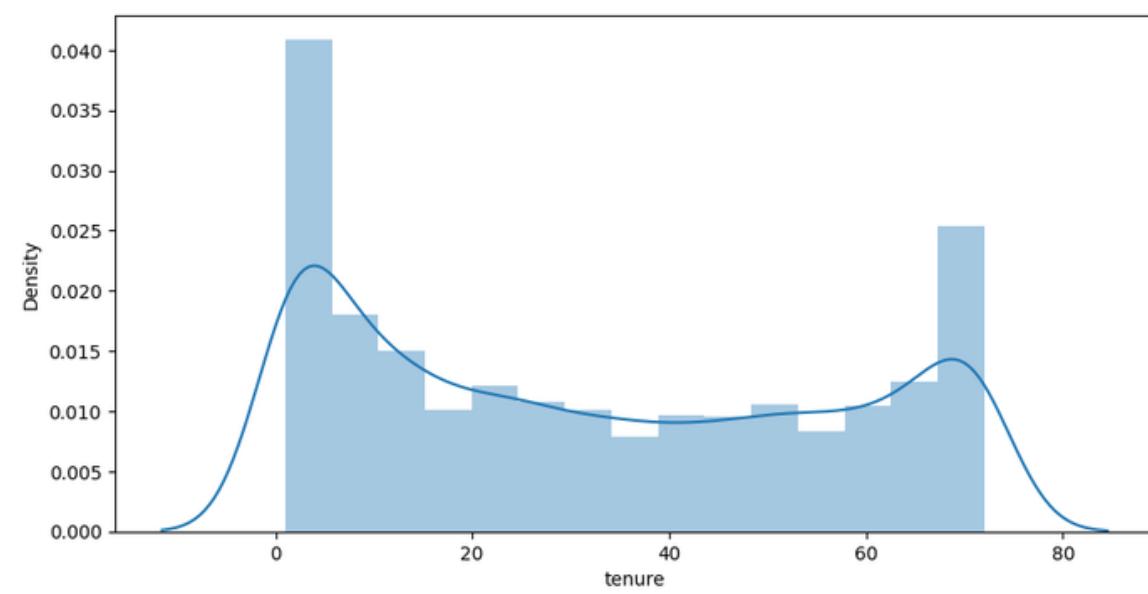
f,ax = plt.subplots(figsize=(10,5))
sns.distplot(df['MonthlyCharges'])

f,ax = plt.subplots(figsize=(10,5))
sns.distplot(df['TotalCharges'])
```

Penjelasan:

Menggunakan rumus sns.distplot() = berfungsi untuk menampilkan visualisasi data numerik dalam bentuk histogram

Output:



Penjelasan:

Dari ketiga tampilan grafik histogram tersebut, dapat diartikan bahwa tidak terdapat value yang memiliki gap/perbedaan yang jauh dari value lainnya

Langkah 3 : Mencari anomali handling menggunakan boxplot

Code:

```
plt.figure()  
sns.boxplot(x=df['tenure'])  
plt.show()  
plt.figure()  
sns.boxplot(x=df['MonthlyCharges'])  
plt.show()  
plt.figure()  
sns.boxplot(x=df['TotalCharges'])  
plt.show()
```

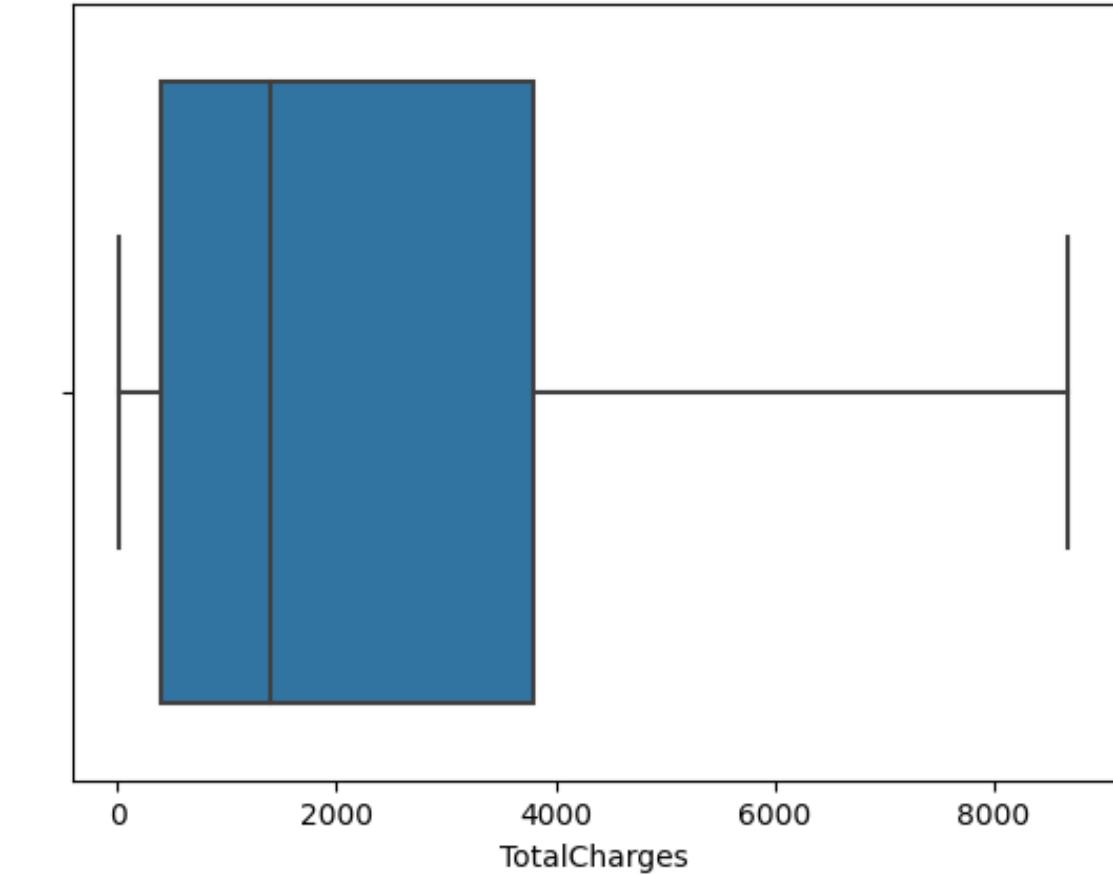
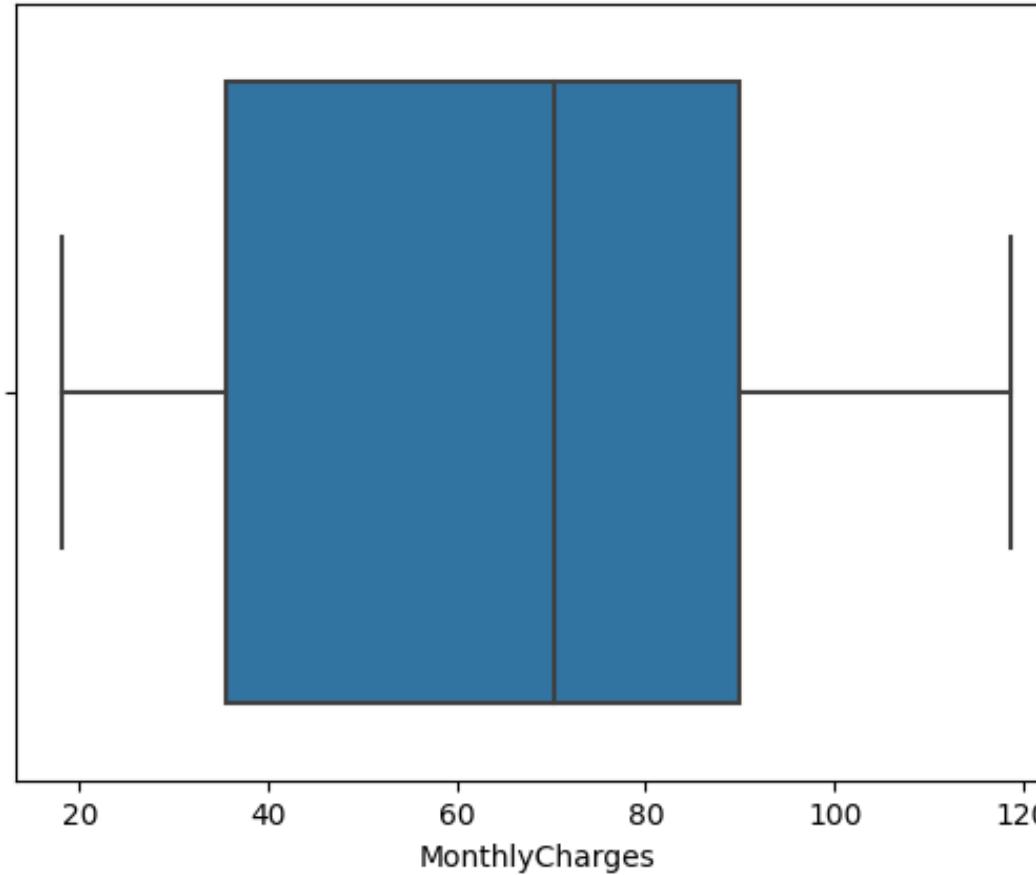
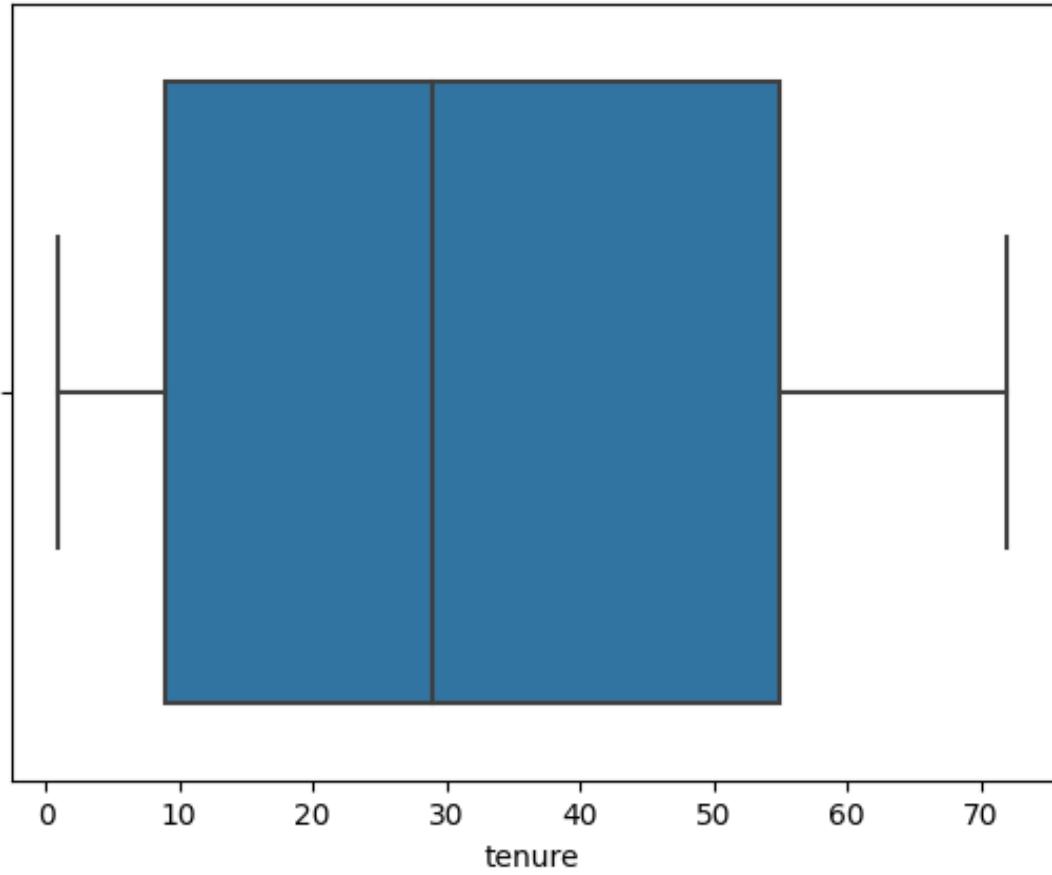
Penjelasan:

Menggunakan rumus plt.figure() = berfungsi untuk mengolah visualisasi data yang diinginkan

Menggunakan rumus plt.show() = berfungsi untuk menampilkan visualisasi data yang diinginkan

dapat terlihat melalui boxplot tersebut bahwa tidak ada data yang outliers

Output:



Penjelasan:

Dari ketiga tampilan boxplot tersebut, dapat diartikan bahwa tidak terdapat value yang memiliki gap/perbedaan yang jauh dari value lainnya

LINK GOOGLE COLAB

Topik 5 - Daffa Andika Zain

&

Topik 6 - Daffa Andika Zain

Thank You