# Data Cleansing

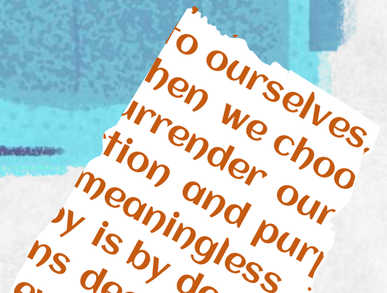
**ZULFA NABILAH NURVITASARI (099)**

**KELAS: DATA ANALYTICS**

# Contents

# Data Cleansing

Data Cleansing sangat diperlukan dalam mengolah data agar informasi yang didapatkan dari hasil menganalis data tersebut akurat dan dapat membantu untuk mengambil keputusan yang tepat sesuai dengan permasalahan yang terjadi. Dalam proses data cleansing kia dapat menggunakan Pandasy Python pada Google Colab. Berikut merupakan langkah-langkahnya.



# Import Package

Langkah pertama yang harus dilakukan yaitu mengimport package untuk mengaktifkan library sehingga function-function yang diperlukan dalam proses data cleansing nanti dapat dengan mudah digunakan.

```
!pip install pandas

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: pandas in /usr/local/lib/python3.9/dist-packages (1.4.4)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.9/dist-packages (from pandas) (20
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.9/dist-packages (from p
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.9/dist-packages (from pandas) (1
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.9/dist-packages (from python-dateutil
```

```
import warnings
warnings.filterwarnings('ignore')

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

warnings. filterwarnings('ignore') digunakan untuk menghiraukan warning dari code yang dibuat. Numpy merupakan library yang digunakan untuk data yang berhubungan dengan array, numpy menyediakan fungsi yang dapat digunakan untuk melakukan perhitungan saintifik seperti statistik, aljabar, matriks, dsb. Numpy di sini didefinisikan sebagai np. Pandas merupakan library yang digunakan untuk memproses data, menganalisis data, membersihkan data, dan memanipulasi data dalam data frame. Pandas di sini didefinisikan sebagai pd. Seaborn dan matplotlib.pyplot merupakan library yang digunakan untuk visualisasi data. Di sini seaborn didefinisikan sebagai sns, sedangkan matplotlib.pyplot didefinisikan sebagai plt. Semua library tersebut dipanggil dengan function import.

# Load Dataset

```
[ ]  from google.colab import files
     uploaded = files.upload()
```

Choose Files WA_Fn-Us...er-Churn.csv

- **WA_Fn-UseC_-Telco-Customer-Churn.csv**(text/csv) - 977501 bytes, last modified: 4/9/2023 - 100% done

```
Saving WA_Fn-UseC_-Telco-Customer-Churn.csv to WA_Fn-UseC_-Telco-Customer-Churn (1).csv
```

Untuk mengimport file dari penyimpanan laptop ke google colab diperlukan code from google.colab import files. Definisikan uploaded sebagai file yang diupload menggunakan function upload(). Dapat dilihat bahwa file WA_Fn-UseC_-Telco-Customer-Churn.csv telah diupload ke google colab.

# Load Dataset

Untuk memunculkan dataset yang berbentuk csv, maka diperlukan function pd.read_csv('Nama file yang telah diimport ke Google Colab'). Fuction ini didefinisikan sebagai df. Df pada bagian akhir digunakan untuk memunculkan data yang telah didefinisikan sebelumnya.

```
df = pd.read_csv('WA_Fn-UseC_-Telco-Customer-Churn.csv')
df
```

| | customerID | gender | SeniorCitizen | Partner | Dependents | tenure | PhoneService | MultipleLines | InternetSe |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 7590-VHVEG | Female | 0 | Yes | No | 1 | No | No phone service | |
| 1 | 5575-GNVDE | Male | 0 | No | No | 34 | Yes | No | |
| 2 | 3668-QPYBK | Male | 0 | No | No | 2 | Yes | No | |
| 3 | 7795-CFOCW | Male | 0 | No | No | 45 | No | No phone service | |

# Missing Value Checking

Dengan menggunakan function info() kita dapat mengetahui informasi mengenai variabel/atribut yang ada pada dataset, atribut mana yang memiliki missing value, dan tipe data pada setiap atribut, jumlah kolom, dan juga jumlah data yang dimasukkan. Dapat dilihat bahwa jumlah data/pelanggan yang dimasukkan pada dataset ialah sebanyak 7043 orang. Pada kolom Non-Null Count dapat diketahui bahwa di semua atribut atau kolom terdiri dari 7043 data. Hal ini membuktikan bahwa tidak terdapat missing value pada dataset tersebut.

```
df.info()
```

```
<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
```

# Missing Value Checking

```
df.isnull().sum()
```

```
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
```

Selain menggunakan function info(), kita juga dapat menggunakan function isnull() dalam pengecekan missing value. Dengan function isnull() kita dapat melihat jumlah baris yang mengalami missing value pada setiap kolom/atribut. Dapat dilihat bahwa missing value pada setiap atribut 0. Dapat didimpulkan bahwa semua atribut tidak memiliki missing value.

# Categorical Data Encoding

## 1. Label Encoding

```
#Pengelompokkan data
df["MultipleLines"] = df["MultipleLines"].replace("No phone service","No")
df["InternetService"] = df["InternetService"].replace(["DSL", "Fiber optic"],"Yes")
df["OnlineSecurity"] = df["OnlineSecurity"].replace("No internet service","No")
df["OnlineBackup"] = df["OnlineBackup"].replace("No internet service","No")
df["DeviceProtection"] = df["DeviceProtection"].replace("No internet service","No")
df["TechSupport"] = df["TechSupport"].replace("No internet service","No")
df["StreamingTV"] = df["StreamingTV"].replace("No internet service","No")
df["StreamingMovies"] = df["StreamingMovies"].replace("No internet service","No")
df
```

| | customerID | gender | SeniorCitizen | Partner | Dependents | tenure | PhoneService | MultipleLines |
|---|---|---|---|---|---|---|---|---|
| 0 | 7590-VHVEG | Female | 0 | Yes | No | 1 | No | No |
| 1 | 5575-GNVDE | Male | 0 | No | No | 34 | Yes | No |
| 2 | 3668-QPYBK | Male | 0 | No | No | 2 | Yes | No |

Atribut-atribut tersebut memiliki 3 kategori. Oleh karena itu, dilakukan pengelompokan menjadi 2 kategori yaitu "yes" dan "no" dengan menggunakan function df["nama kolom"].replace("value lama","value baru") untuk mengubah value pada kolom yang akan dikelompokkan agar memudahkan pada saat melakukan pengolahan data. Misalnya saja untuk mencari rata-rata, agar terdapat jawaban yang pasti antara "yes" dan "no", maka diharapkan hasil dari perhitungan mean values yang telah diidentifikasikan dengan angka numerik tersebut bukan berupa desimal. Pada kolom MultipleLines, "No phone service" dikelompokkan menjadi "No". "DSL" dan "Fiber optic" pada kolom InternetService dikelompokkan menjadi "Yes". Sedangkan, "No internet service" pada kolom OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, dan StreamingMovies dikelompokkan menjadi "No".

# Categorical Data Encoding

```
df['gender'] = df['gender'].astype('category').cat.codes
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['MultipleLines'] = df['MultipleLines'].astype('category').cat.codes
df['InternetService'] = df['InternetService'].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.head(3)
```

| | customerID | gender | SeniorCitizen | Partner | Dependents | tenure | PhoneService | MultipleLines | InternetService |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 7590-VHVEG | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 5575-GNVDE | 1 | 0 | 0 | 0 | 34 | 1 | 0 | 1 |
| 2 | 3668-QPYBK | 1 | 0 | 0 | 0 | 2 | 1 | 0 | 1 |

3 rows × 21 columns

Label encoding diterapkan pada atribut gender, Partner, Dependents, PhoneService, PaperlessBilling, Churn, MultipleLines, InternetService, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, dan StreamingMovies karena data yang bersifat kategori tidak bisa diolah/diproses lebih dalam lagi sehingga diperlukan manipulasi data untuk membuat seakan-akan data yang dimasukkan berupa numerik. Label encoding digunakan pada ke-6 atribut tersebut karena kategori tidak terlalu banyak (hanya terdiri dari 2 macam kategori) sehingga memudahkan pada saat mengubah data kategori menjadi numerik. Pada atribut gender, 0 merupakan 'Female', sedangkan 1 merupakan 'Male'. Untuk atribut Partner, Dependents, PhoneService, PaperlessBilling, Churn, MultipleLines, InternetService, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, dan StreamingMovies 0 merupakan 'No', sedangkan 1 merupakan 'Yes'.

# Categorical Data Encoding

## 2. One Hot Encoding

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

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

One hot encoding pada atribut PaymentMethod dilakukan dengan menggunakan function pd.get.dummies. One hot encoding mempresentasikan data dengan 0 dan 1 dimana pada saat terdapat baris yang mengandung nilai suatu kategori tersebut bernilai 1 dan sisanya 0.

# Categorical Data Encoding

untuk menggabungkan df dengan data one hot encoding yang telah dilakukan, maka diperlukan code df = pd.concat([df, dummies_PaymentMethod], axis=1) yang didefinisikan sebagai df. Dapat dilihat terdapat 4 kolom baru pada df, yaitu kolom dari hasil one hot encoding yang telah dilakukan pada atribut PaymentMethod.

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

| s | TotalCharges | Churn | PaymentMethod_Bank transfer (automatic) | PaymentMethod_Credit card (automatic) | PaymentMethod_Electronic check | PaymentMethod_Mailed check |
|---|---|---|---|---|---|---|
| 5 | 29.85 | 0 | 0 | 0 | 1 | 0 |
| 5 | 1889.5 | 0 | 0 | 0 | 0 | 1 |
| 5 | 108.15 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1840.75 | 0 | 1 | 0 | 0 | 0 |
| 0 | 151.65 | 1 | 0 | 0 | 1 | 0 |

# Categorical Data Encoding

## 3. Frequency Encoding

```
freq_Contract = df['Contract'].value_counts().reset_index()
freq_Contract.rename(columns={"index": "Contract", "Contract": "freq_Contract"}, inplace = True)
freq_Contract
```

| | Contract | freq_Contract |
|---|---|---|
| 0 | Month-to-month | 3875 |
| 1 | Two year | 1695 |
| 2 | One year | 1473 |

Pada dataContract tidak dapat dilakukan label encoding karena terdapat lebih dari dua kategori dan tidak bisa dikelompokkan menjadi ukuran yang lebih kecil lagi (2 kategori) seperti MultipleLines, InternetServise, dan lainnya. pada freq encoding ini setiap baris data yang bersifat kategorik akan diisi dengan jumlah kemunculan data pada atribut tersebut. Untuk itu, diperlukan code df['Contract'].value_counts().reset_index() untuk menghitung jumlah setiap data yang muncul pada kolom Contract yang didefinisikan sebagai freq_Contract. Untuk melihat hasil dari perhitungan tersebut, dibutuhkan code freq_Contract.rename(columns={"index": "Contract", "Contract": "freq_Contract"}, inplace = True) pada kolom 1 index diubah menjadi kategori yang ada pada kolom Contract agar terdapat irisan antara freq_Contract dengan df ketika digabungkan. Data pada Contract juga diubah menjadi freq_Contract untuk melihat jumlah frekuensi data yang muncul. freq_Contract pada bagian akhir digunakan untuk memunculkan freq_Contract yang telah diidentifikasikan.

# Categorical Data Encoding

```
df = df.merge(freq_Contract[['Contract','freq_Contract']], on='Contract', how='inner')
df.head()
```

| Churn | PaymentMethod_Bank transfer (automatic) | PaymentMethod_Credit card (automatic) | PaymentMethod_Electronic check | PaymentMethod_Mailed check | freq_Contract |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 3875 |
| 1 | 0 | 0 | 0 | 1 | 3875 |
| 1 | 0 | 0 | 1 | 0 | 3875 |
| 1 | 0 | 0 | 1 | 0 | 3875 |
| 0 | 0 | 1 | 0 | 0 | 3875 |

Untuk memasukan freq_Contract pada data frame df maka diperlukan code df.merge(freq_Contract[['Contract','freq_Contract']], on='Contract', how='inner') yang bertujuan untuk menggabungkan secara inner (inner join) freq_Contract yang terdiri dari Contract dan freq_Contract dengan df yang juga mengandung Contract menggunakan function merge. df.head() pada bagian akhir digunakan untuk memunculkan data 5 baris teratas dari df. Dapat dilihat bahwa terdapat kolom baru, yaitu freq_Contract.

# Anomalies and Outlier Handling

Sebelum melakukan penanganan terhadap data anomali dan outlier dapat dilihat terlebih dahulu tipe data dengan function df.info() dibagian atas bahwa kolom TotalCharges bertipe data objek, seharusnya tipe data dari TotalCharges adalah float. Oleh karena itu, dilakukan perubahan tipe datateriebih dahulu pada kolom tersebut.

```
df['TotalCharges'] = 0
df['TotalCharges'] = df['TotalCharges'].astype('float')
df['TotalCharges'] = df['tenure']*df['MonthlyCharges']
df.head(3)
```

| ort | StreamingTV | StreamingMovies | Contract | PaperlessBilling | PaymentMethod | MonthlyCharges | TotalCharges |
|---|---|---|---|---|---|---|---|
| No | No | No | Month-to-month | Yes | Electronic check | 29.85 | 29.85 |
| No | No | No | One year | No | Mailed check | 56.95 | 1936.30 |
| No | No | No | Month-to-month | Yes | Mailed check | 53.85 | 107.70 |

```
16  PaperlessBilling  7043 non-null   object
17  PaymentMethod     7043 non-null   object
18  MonthlyCharges    7043 non-null   float64
19  TotalCharges      7043 non-null   float64
20  Churn             7043 non-null   object
```

Perubahan tipe data TotalCharges dilakukan dengan mengubah semua value TotalCharges menjadi 0 dan mengganti tipe datanya menjadi float, lalu valuenya diisi kembali dengan hasil dari perkalian antara tenure dan MonthlyCharges. Dapat diketahui dengan function info() bahwa tipe data TotalCharges telah berubah menjadi float.

# Anomalies and Outlier Handling

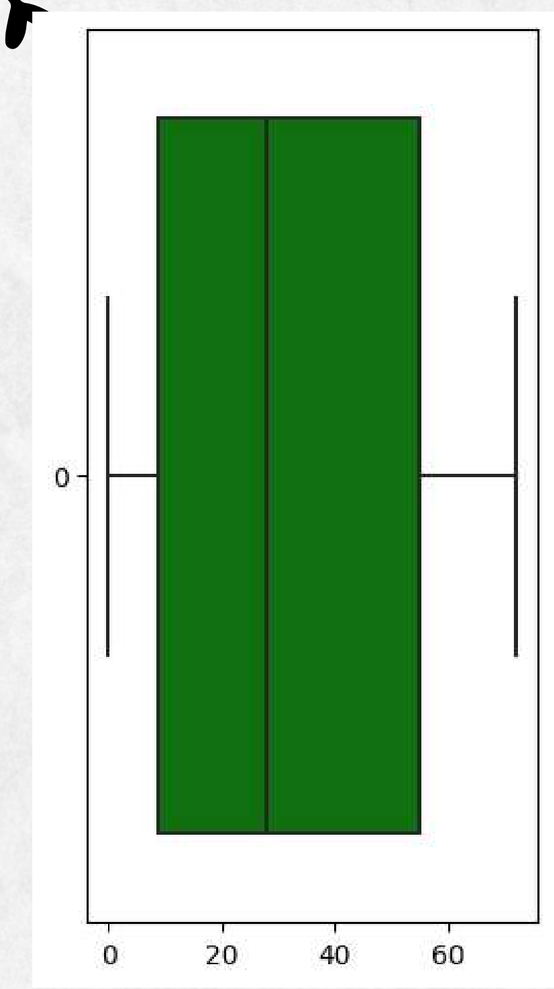
```
df[['tenure', 'MonthlyCharges','TotalCharges']].describe()
```

| | tenure | MonthlyCharges | TotalCharges |
|---|---|---|---|
| count | 7043.000000 | 7043.000000 | 7043.000000 |
| mean | 32.371149 | 64.761692 | 2279.581350 |
| std | 24.559481 | 30.090047 | 2264.729447 |
| min | 0.000000 | 18.250000 | 0.000000 |
| 25% | 9.000000 | 35.500000 | 394.000000 |
| 50% | 29.000000 | 70.350000 | 1393.600000 |
| 75% | 55.000000 | 89.850000 | 3786.100000 |
| max | 72.000000 | 118.750000 | 8550.000000 |

Data yang digunakan merupakan data yang numerik sehingga hanya atribut tenur, MonthlyCharges, dan TotalCharges saja yang digunakan. df[['tenure', 'MonthlyCharges','TotalCharges']].describe() digunakan untuk melihat statistik deskriptif pada data frame df dari kolom tenur, MonthlyCharges, dan TotalCharges. Dari data yang ditampilkan di atas dapat diketahui bahwa terdapat perbedaan yang besar antara nilai mean dengan median pada kolom TotalCharges. Hal ini menunjukan kemungkinan adanya anomali atau outlier pada data. Oleh karena itu, perlu dilakukan pengecekan melalui boxplot.

# Anomalies and Outlier Handling

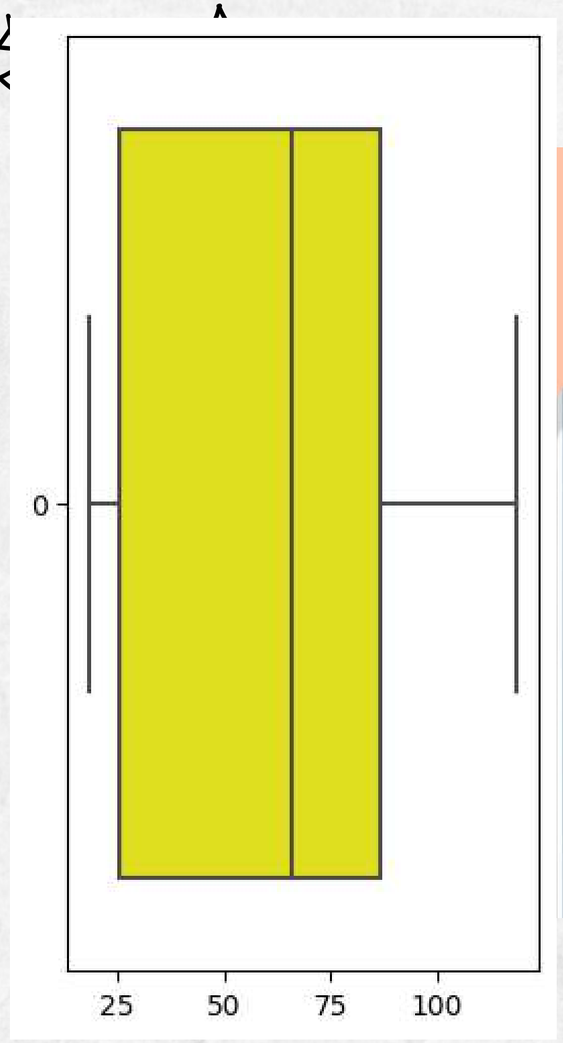
```
plt.figure(figsize = (3,6))
sns.boxplot(df['tenure'], color = 'green', orient = 'h')
```




Code plt.figure(figsize = (3,6)) digunakan untuk menentukan ukuran boxplot yang akan ditampilkan. Untuk membuat boxplot dari data tenure memerlukan function boxplot dari library seaborn yang didefinisikan sebagai sns dengan code sns.boxplot(df['tenure'], color = 'green', orient = 'h'). Boxplot data tenure ditampilkan dengan posisi horizontal dan berwarna hijau. Dapat dikatakan bahwa boxplot berbentuk simetris sehingga dapat dikatakan bahwa data tenure berdistribusi normal. Selain itu, tidak terdapat data outlier pada boxplot tersebut sehingga tidak diperlukan penanganan lebih lanjut. Jika terdapat outlier maka dapat ditangani dengan transformasi log, Z-score, dan IQR.

# Anomalies and Outlier Handling

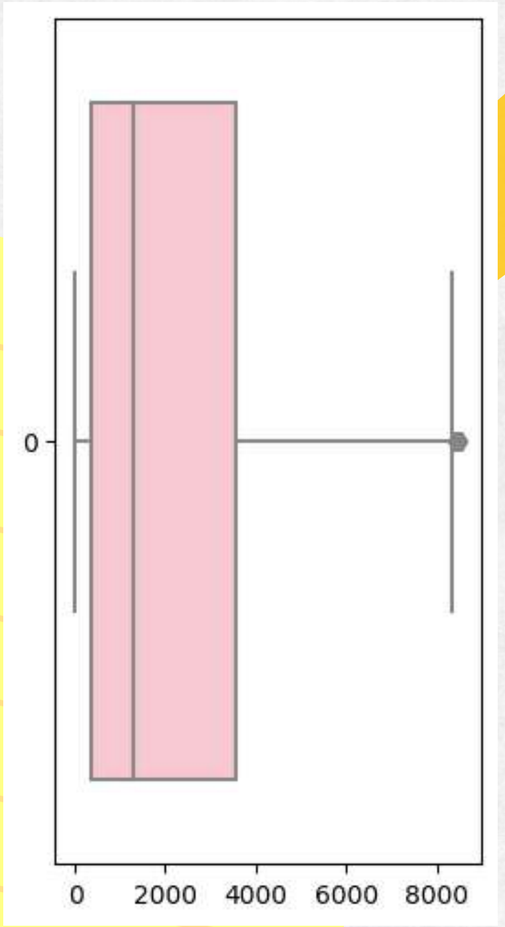
```
plt.figure(figsize = (3,6))
sns.boxplot(df['MonthlyCharges'], color = 'yellow', orient = 'h')
```




Code plt.figure(figsize = (3,6)) digunakan untuk menentukan ukuran boxplot yang akan ditampilkan. Untuk membuat boxplot dari data MonthlyCharges memerlukan function boxplot dari library seaborn yang didefinisikan sebagai sns dengan code sns.boxplot(df['MonthlyCharges'], color = 'yellow', orient = 'h'). Boxplot data MonthlyCharges ditampilkan dengan posisi horizontal dan berwarna kuning. Dapat dilihat bahwa boxplot berbentuk hampir simetris sehingga dapat dikatakan bahwa data MonthlyCharges berdistribusi normal. Selain itu, tidak terdapat data outlier pada boxplot tersebut sehingga tidak diperlukan penanganan lebih lanjut. Jika terdapat outlier maka dapat ditangani dengan transformasi log, Z-score, dan IQR.

# Anomalies and Outlier Handling

```
plt.figure(figsize = (3,6))
sns.boxplot(df['TotalCharges'], color = 'pink', orient = 'h')
```




Code plt.figure(figsize = (3,6)) digunakan untuk menentukan ukuran boxplot yang akan ditampilkan. Untuk membuat boxplot dari data TotalCharges memerlukan function boxplot dari library seaborn yang didefinisikan sebagai sns dengan code sns.boxplot(df['TotalCharges'], color = 'pink', orient = 'h'). Boxplot data TotalCharges ditampilkan dengan posisi horizontal dan berwarna pink. Dapat dilihat bahwa tidak terdapat data outlier pada boxplot tersebut sehingga tidak diperlukan penanganan lebih lanjut. Jika terdapat outlier maka dapat ditangani dengan transformasi log, Z-score, dan IQR.

# Link Google Colab

Data Cleansing:

https://colab.research.google.com/drive/1lCdZs47n120xWm9Xe2vTzEK4qJaYu6Vg#scrollTo=pd6h8aQkppSn

Topik 6:

https://colab.research.google.com/drive/1wX6wMT_ZOKlJrMuixHVdPRQQ6h0u-M39#scrollTo=381bfdde-9ffc-4b13-aa4d-4ada2e022e19

# Terima Kasih