TUGAS LAB IF540 MACHINE LEARNING

WEEK 02 : Data Preprocessing

Semester Genap 2022/2023

In [1]:

```python
# Run this code when you restart the machine
# Fill in with YOUR name and NIM
import datetime
import uuid

myName = "Christopher Darren"
myNIM = "00000054804"
```

In [2]:

```python
myDate = datetime.datetime.now()
myDevice = str(uuid.uuid1())

print("Name: \t\t{}".format(myName))
print("NIM: \t\t{}".format(myNIM))
print("Start: \t\t{}".format(myDate))
print("Device ID: \t{}".format(myDevice))
```

```
Name:           Christopher Darren
NIM:            00000054804
Start:          2023-02-16 20:41:16.786311
Device ID:      9647214d-adff-11ed-b734-f02f74a116e8
```

Dataset yang dipakai:

1. Vaccination data – sumber : https://www.kaggle.com/datasets/umeshkumar017/vaccination-data
2. Fuel Consumption – sumber : https://www.kaggle.com/datasets/sarita19/fuel-consumption

---

Hasil kerja

Importing system library

In [102…

```python
from IPython.display import Image
%matplotlib Inline
```

In [103…

```python
import pandas as pd
from io import StringIO
import sys
```

In [104…

```python
#reading data
df= pd.read_csv(r"D:\SEMESTER 4\IF540 Machine Learning\LAB\week2\vaccination_data.csv")
df.head(5)
```

Out[104]:

	COUNTRY	ISO3	WHO_REGION	DATA_SOURCE	DATE_UPDATED	TOTAL_VACCINATIONS	PERSONS_VACCINATED_1PLUS_DOSE	TOTA
0	Afghanistan	AFG	EMRO	REPORTING	2022-07-19	7885045.0	7139453.0	
1	Albania	ALB	EURO	REPORTING	2022-07-24	2934116.0	1330520.0	
2	Algeria	DZA	AFRO	REPORTING	2022-07-03	15205854.0	7840131.0	
3	American Samoa	ASM	WPRO	REPORTING	2022-06-24	109507.0	44586.0	
4	Andorra	AND	EURO	REPORTING	2022-07-10	153531.0	57888.0	

In [105…

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 229 entries, 0 to 228
Data columns (total 16 columns):
 #   Column                               Non-Null Count  Dtype  
---  ------                               --------------  -----  
 0   COUNTRY                              229 non-null    object 
 1   ISO3                                 229 non-null    object 
 2   WHO_REGION                           229 non-null    object 
 3   DATA_SOURCE                          229 non-null    object 
 4   DATE_UPDATED                         229 non-null    object 
 5   TOTAL_VACCINATIONS                   228 non-null    float64
 6   PERSONS_VACCINATED_1PLUS_DOSE        228 non-null    float64
 7   TOTAL_VACCINATIONS_PER100            228 non-null    float64
 8   PERSONS_VACCINATED_1PLUS_DOSE_PER100  228 non-null    float64
 9   PERSONS_FULLY_VACCINATED             228 non-null    float64
 10  PERSONS_FULLY_VACCINATED_PER100      228 non-null    float64
 11  VACCINES_USED                        225 non-null    object 
 12  FIRST_VACCINE_DATE                   207 non-null    object 
 13  NUMBER_VACCINES_TYPES_USED           225 non-null    float64
 14  PERSONS_BOOSTER_ADD_DOSE             205 non-null    float64
 15  PERSONS_BOOSTER_ADD_DOSE_PER100      205 non-null    float64
dtypes: float64(9), object(7)
memory usage: 28.8+ KB
```

In [106…

```python
df.dtypes
```

Out[106]:

```
COUNTRY                                  object
ISO3                                     object
WHO_REGION                               object
DATA_SOURCE                              object
DATE_UPDATED                             object
TOTAL_VACCINATIONS                      float64
PERSONS_VACCINATED_1PLUS_DOSE           float64
TOTAL_VACCINATIONS_PER100               float64
PERSONS_VACCINATED_1PLUS_DOSE_PER100    float64
PERSONS_FULLY_VACCINATED                float64
PERSONS_FULLY_VACCINATED_PER100         float64
VACCINES_USED                            object
FIRST_VACCINE_DATE                       object
NUMBER_VACCINES_TYPES_USED              float64
PERSONS_BOOSTER_ADD_DOSE                float64
PERSONS_BOOSTER_ADD_DOSE_PER100         float64
dtype: object
```

In [107…

```python
df.shape
```

Out[107]:

```
(229, 16)
```

In [108…

```python
#Access the underlying numpy array
df.values
```

Out[108]:

```
array([['Afghanistan', 'AFG', 'EMRO', ..., 11.0, nan, nan],
       ['Albania', 'ALB', 'EURO', ..., 5.0, 338290.0, 11.887],
       ['Algeria', 'DZA', 'AFRO', ..., 4.0, 514063.0, 1.172],
       ...,
       ['Yemen', 'YEM', 'EMRO', ..., 11.0, 80.0, 0.0],
       ['Zambia', 'ZMB', 'AFRO', ..., 3.0, 428303.0, 2.33],
       ['Zimbabwe', 'ZWE', 'AFRO', ..., 4.0, 908996.0, 6.116]],
      dtype=object)
```

Eliminating Samples of Features With Missing Values

One of the easiest ways to deal with missing data is to simply remove the corresponding features (columns) or samples (rows) from the dataset entirely; rows with missing values can be easily dropped via the dropna methods

In [109…

```python
df.dropna(axis=0)
```

Out[109]:

	COUNTRY	ISO3	WHO_REGION	DATA_SOURCE	DATE_UPDATED	TOTAL_VACCINATIONS	PERSONS_VACCINATED_1PLUS_DOSE	TOT
1	Albania	ALB	EURO	REPORTING	2022-07-24	2934116.0	1330520.0	
2	Algeria	DZA	AFRO	REPORTING	2022-07-03	15205854.0	7840131.0	
3	American Samoa	ASM	WPRO	REPORTING	2022-06-24	109507.0	44586.0	
4	Andorra	AND	EURO	REPORTING	2022-07-10	153531.0	57888.0	
5	Angola	AGO	AFRO	REPORTING	2022-07-17	21099865.0	13507932.0	
...	...	...	...	...	...	...	...	
224	Viet Nam	VNM	WPRO	REPORTING	2022-07-07	234856999.0	86785069.0	
225	Wallis and Futuna	WLF	WPRO	REPORTING	2022-04-14	16426.0	6592.0	
226	Yemen	YEM	EMRO	REPORTING	2022-07-04	874886.0	708152.0	
227	Zambia	ZMB	AFRO	REPORTING	2022-07-10	7409521.0	6649681.0	
228	Zimbabwe	ZWE	AFRO	REPORTING	2022-07-17	11928290.0	6333666.0	

192 rows × 16 columns

In [110…

```
#checking is null or not
df.isnull().sum()
```

Out[110]:

```
COUNTRY                              0
ISO3                                 0
WHO_REGION                           0
DATA_SOURCE                          0
DATE_UPDATED                         0
TOTAL_VACCINATIONS                   1
PERSONS_VACCINATED_1PLUS_DOSE        1
TOTAL_VACCINATIONS_PER100            1
PERSONS_VACCINATED_1PLUS_DOSE_PER100 1
PERSONS_FULLY_VACCINATED             1
PERSONS_FULLY_VACCINATED_PER100      1
VACCINES_USED                        4
FIRST_VACCINE_DATE                  22
NUMBER_VACCINES_TYPES_USED           4
PERSONS_BOOSTER_ADD_DOSE            24
PERSONS_BOOSTER_ADD_DOSE_PER100     24
dtype: int64
```

In [111…

```
# remove columns that contain missing values
df.dropna(axis=1)
```

Out[111]:

	COUNTRY	ISO3	WHO_REGION	DATA_SOURCE	DATE_UPDATED
0	Afghanistan	AFG	EMRO	REPORTING	2022-07-19
1	Albania	ALB	EURO	REPORTING	2022-07-24
2	Algeria	DZA	AFRO	REPORTING	2022-07-03
3	American Samoa	ASM	WPRO	REPORTING	2022-06-24
4	Andorra	AND	EURO	REPORTING	2022-07-10
...	...	...	...	...	...
224	Viet Nam	VNM	WPRO	REPORTING	2022-07-07
225	Wallis and Futuna	WLF	WPRO	REPORTING	2022-04-14
226	Yemen	YEM	EMRO	REPORTING	2022-07-04
227	Zambia	ZMB	AFRO	REPORTING	2022-07-10
228	Zimbabwe	ZWE	AFRO	REPORTING	2022-07-17

229 rows × 5 columns

Imputing missing values

In [112…

```
#again : our original array
df.values
```

Out[112]:

```
array([['Afghanistan', 'AFG', 'EMRO', ..., 11.0, nan, nan],
       ['Albania', 'ALB', 'EURO', ..., 5.0, 338290.0, 11.887],
       ['Algeria', 'DZA', 'AFRO', ..., 4.0, 514063.0, 1.172],
       ...,
       ['Yemen', 'YEM', 'EMRO', ..., 11.0, 80.0, 0.0],
       ['Zambia', 'ZMB', 'AFRO', ..., 3.0, 428303.0, 2.33],
       ['Zimbabwe', 'ZWE', 'AFRO', ..., 4.0, 908996.0, 6.116]],
      dtype=object)
```

Importing numpy library

In [113…

```
from sklearn.impute import SimpleImputer
import numpy as np

imr = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
imr = imr.fit(df.values)
imputed_data = imr.transform(df.values)
imputed_data
```

Out[113]:

```
array([['Afghanistan', 'AFG', 'EMRO', ..., 11.0, 0.0, 0.0],
       ['Albania', 'ALB', 'EURO', ..., 5.0, 338290.0, 11.887],
       ['Algeria', 'DZA', 'AFRO', ..., 4.0, 514063.0, 1.172],
       ...,
       ['Yemen', 'YEM', 'EMRO', ..., 11.0, 80.0, 0.0],
       ['Zambia', 'ZMB', 'AFRO', ..., 3.0, 428303.0, 2.33],
       ['Zimbabwe', 'ZWE', 'AFRO', ..., 4.0, 908996.0, 6.116]],
      dtype=object)
```

In [114…

```
df.fillna(df.mean())
```

```
C:\Users\Darren\AppData\Local\Temp\ipykernel_20836\634187881.py:1: FutureWarning: Dropping of nuisance columns
in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError
.  Select only valid columns before calling the reduction.
  df.fillna(df.mean())
```

Out[114]:

	COUNTRY	ISO3	WHO_REGION	DATA_SOURCE	DATE_UPDATED	TOTAL_VACCINATIONS	PERSONS_VACCINATED_1PLUS_DOSE	TO
0	Afghanistan	AFG	EMRO	REPORTING	2022-07-19	7885045.0	7139453.0	
1	Albania	ALB	EURO	REPORTING	2022-07-24	2934116.0	1330520.0	
2	Algeria	DZA	AFRO	REPORTING	2022-07-03	15205854.0	7840131.0	
3	American Samoa	ASM	WPRO	REPORTING	2022-06-24	109507.0	44586.0	
4	Andorra	AND	EURO	REPORTING	2022-07-10	153531.0	57888.0	
...	...	...	...	...	...	...	...	
224	Viet Nam	VNM	WPRO	REPORTING	2022-07-07	234856999.0	86785069.0	
225	Wallis and Futuna	WLF	WPRO	REPORTING	2022-04-14	16426.0	6592.0	
226	Yemen	YEM	EMRO	REPORTING	2022-07-04	874886.0	708152.0	
227	Zambia	ZMB	AFRO	REPORTING	2022-07-10	7409521.0	6649681.0	
228	Zimbabwe	ZWE	AFRO	REPORTING	2022-07-17	11928290.0	6333666.0	

229 rows × 16 columns

Handling Categorical Data

In [115…

```
#reading data
df1= pd.read_csv(r"D:\SEMESTER 4\IF540 Machine Learning\LAB\week2\FuelConsumption.csv",index_col=0)
```

```
df1.head(5)
```

Out[115]:

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELCONSUMPTION_CITY	FUEL
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9	
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2	
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0	
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7	
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1	

In [116…

```
df1.tail(10)
```

Out[116]:

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELCONSUMPTION_CITY	FU
1057	2014	VOLVO	S60 AWD	COMPACT	2.5	5	AS6	X	11.6	
1058	2014	VOLVO	S60 AWD	COMPACT	3.0	6	AS6	X	13.2	
1059	2014	VOLVO	S80	MID-SIZE	3.2	6	AS6	X	11.9	
1060	2014	VOLVO	S80 AWD	MID-SIZE	3.0	6	AS6	X	13.2	
1061	2014	VOLVO	XC60	SUV - SMALL	3.2	6	AS6	X	13.0	
1062	2014	VOLVO	XC60 AWD	SUV - SMALL	3.0	6	AS6	X	13.4	
1063	2014	VOLVO	XC60 AWD	SUV - SMALL	3.2	6	AS6	X	13.2	
1064	2014	VOLVO	XC70 AWD	SUV - SMALL	3.0	6	AS6	X	13.4	
1065	2014	VOLVO	XC70 AWD	SUV - SMALL	3.2	6	AS6	X	12.9	
1066	2014	VOLVO	XC90 AWD	SUV - STANDARD	3.2	6	AS6	X	14.9	

In [117…

```
df1.dropna(axis=0)
```

Out[117]:

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELCONSUMPTION_CITY	F
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9	
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2	
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0	
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7	
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1	
...	...	...	...	...	...	...	...	...	...	
1062	2014	VOLVO	XC60 AWD	SUV - SMALL	3.0	6	AS6	X	13.4	
1063	2014	VOLVO	XC60 AWD	SUV - SMALL	3.2	6	AS6	X	13.2	
1064	2014	VOLVO	XC70 AWD	SUV - SMALL	3.0	6	AS6	X	13.4	
1065	2014	VOLVO	XC70 AWD	SUV - SMALL	3.2	6	AS6	X	12.9	
1066	2014	VOLVO	XC90 AWD	SUV - STANDARD	3.2	6	AS6	X	14.9	

1067 rows × 13 columns

In [118…

```
df1.isnull().sum()
```

Out[118]:

```
MODELYEAR                   0
MAKE                        0
MODEL                       0
VEHICLECLASS                0
ENGINESIZE                  0
CYLINDERS                   0
TRANSMISSION                0
FUELTYPE                    0
FUELCONSUMPTION_CITY        0
FUELCONSUMPTION_HWY         0
FUELCONSUMPTION_COMB        0
FUELCONSUMPTION_COMB_MPG    0
CO2EMISSIONS                0
dtype: int64
```

In [119…

```
df1.dropna(axis=1)
```

Out[119]:

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELCONSUMPTION_CITY	F
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9	
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2	
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0	
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7	
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1	
...	...	...	...	...	...	...	...	...	...	
1062	2014	VOLVO	XC60 AWD	SUV - SMALL	3.0	6	AS6	X	13.4	
1063	2014	VOLVO	XC60 AWD	SUV - SMALL	3.2	6	AS6	X	13.2	
1064	2014	VOLVO	XC70 AWD	SUV - SMALL	3.0	6	AS6	X	13.4	
1065	2014	VOLVO	XC70 AWD	SUV - SMALL	3.2	6	AS6	X	12.9	
1066	2014	VOLVO	XC90 AWD	SUV - STANDARD	3.2	6	AS6	X	14.9	

1067 rows × 13 columns

In [120…

```
df1.iloc[:50,]
```

Out[120]:

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELCONSUMPTION_CITY
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1
5	2014	ACURA	RLX	MID-SIZE	3.5	6	AS6	Z	11.9
6	2014	ACURA	TL	MID-SIZE	3.5	6	AS6	Z	11.8
7	2014	ACURA	TL AWD	MID-SIZE	3.7	6	AS6	Z	12.8
8	2014	ACURA	TL AWD	MID-SIZE	3.7	6	M6	Z	13.4
9	2014	ACURA	TSX	COMPACT	2.4	4	AS5	Z	10.6
10	2014	ACURA	TSX	COMPACT	2.4	4	M6	Z	11.2
11	2014	ACURA	TSX	COMPACT	3.5	6	AS5	Z	12.1
12	2014	ASTON MARTIN	DB9	MINICOMPACT	5.9	12	A6	Z	18.0
13	2014	ASTON MARTIN	RAPIDE	SUBCOMPACT	5.9	12	A6	Z	18.0
14	2014	ASTON MARTIN	V8 VANTAGE	TWO-SEATER	4.7	8	AM7	Z	17.4
15	2014	ASTON MARTIN	V8 VANTAGE	TWO-SEATER	4.7	8	M6	Z	18.1
16	2014	ASTON MARTIN	V8 VANTAGE S	TWO-SEATER	4.7	8	AM7	Z	17.4
17	2014	ASTON MARTIN	V8 VANTAGE S	TWO-SEATER	4.7	8	M6	Z	18.1

18	2014	ASTON MARTIN	VANQUISH	MINICOMPACT	5.9	12	A6	Z	18.0
19	2014	AUDI	A4	COMPACT	2.0	4	AV8	Z	9.9
20	2014	AUDI	A4 QUATTRO	COMPACT	2.0	4	AS8	Z	11.5
21	2014	AUDI	A4 QUATTRO	COMPACT	2.0	4	M6	Z	10.8
22	2014	AUDI	A5 CABRIOLET QUATTRO	SUBCOMPACT	2.0	4	AS8	Z	11.5
23	2014	AUDI	A5 QUATTRO	SUBCOMPACT	2.0	4	AS8	Z	11.5
24	2014	AUDI	A5 QUATTRO	SUBCOMPACT	2.0	4	M6	Z	10.8
25	2014	AUDI	A6 QUATTRO	MID-SIZE	2.0	4	AS8	Z	12.0
26	2014	AUDI	A6 QUATTRO	MID-SIZE	3.0	6	AS8	Z	12.8
27	2014	AUDI	A6 QUATTRO TDI CLEAN DIESEL	MID-SIZE	3.0	6	AS8	D	9.8
28	2014	AUDI	A7 QUATTRO	MID-SIZE	3.0	6	AS8	Z	13.1
29	2014	AUDI	A7 QUATTRO TDI CLEAN DIESEL	MID-SIZE	3.0	6	AS8	D	9.8
30	2014	AUDI	A8	MID-SIZE	3.0	6	AS8	Z	13.1
31	2014	AUDI	A8	MID-SIZE	4.0	8	AS8	Z	13.5
32	2014	AUDI	A8 TDI CLEAN DIESEL	MID-SIZE	3.0	6	AS8	D	10.0
33	2014	AUDI	A8L	FULL-SIZE	3.0	6	AS8	Z	13.1
34	2014	AUDI	A8L	FULL-SIZE	4.0	8	AS8	Z	14.7
35	2014	AUDI	A8L	FULL-SIZE	6.3	12	AS8	Z	18.2
36	2014	AUDI	A8L TDI CLEAN DIESEL	FULL-SIZE	3.0	6	AS8	D	10.1
37	2014	AUDI	ALLROAD QUATTRO	STATION WAGON - SMALL	2.0	4	AS8	Z	11.8
38	2014	AUDI	Q5	SUV - SMALL	2.0	4	AS8	Z	12.0
39	2014	AUDI	Q5	SUV - SMALL	3.0	6	AS8	Z	12.9
40	2014	AUDI	Q5 HYBRID	SUV - SMALL	2.0	4	AS8	Z	9.9
41	2014	AUDI	Q5 TDI CLEAN DIESEL	SUV - SMALL	3.0	6	AS8	D	10.3
42	2014	AUDI	Q7	SUV - STANDARD	3.0	6	AS8	Z	15.1
43	2014	AUDI	Q7 TDI CLEAN DIESEL	SUV - STANDARD	3.0	6	AS8	D	12.9
44	2014	AUDI	R8	TWO-SEATER	4.2	8	A7	Z	17.6
45	2014	AUDI	R8	TWO-SEATER	4.2	8	M6	Z	21.2
46	2014	AUDI	R8	TWO-SEATER	5.2	10	A7	Z	18.8
47	2014	AUDI	R8	TWO-SEATER	5.2	10	M6	Z	21.1
48	2014	AUDI	R8 SPYDER	TWO-SEATER	4.2	8	A7	Z	17.6
49	2014	AUDI	R8 SPYDER	TWO-SEATER	4.2	8	M6	Z	21.2

In [121…

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1067 entries, 0 to 1066
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   MODELYEAR                 1067 non-null   int64  
 1   MAKE                      1067 non-null   object 
 2   MODEL                     1067 non-null   object 
 3   VEHICLECLASS              1067 non-null   object 
 4   ENGINESIZE                1067 non-null   float64
 5   CYLINDERS                 1067 non-null   int64  
 6   TRANSMISSION              1067 non-null   object 
 7   FUELTYPE                  1067 non-null   object 
 8   FUELCONSUMPTION_CITY      1067 non-null   float64
 9   FUELCONSUMPTION_HWY       1067 non-null   float64
 10  FUELCONSUMPTION_COMB      1067 non-null   float64
 11  FUELCONSUMPTION_COMB_MPG  1067 non-null   int64  
 12  CO2EMISSIONS              1067 non-null   int64  
dtypes: float64(4), int64(4), object(5)
memory usage: 116.7+ KB
```

In [122…

```python
import pandas as pd

df1 = pd.DataFrame([['ACURA','Z',3.5,'SUV-SMALL'],
                    ['AUDI','D',3.0,'SUV-STANDARD'],
                    ['BUICK','E',3.6,'MID-SIZE'],
                    ['VOLVO','X',2.0,'COMPACT']])
df1.columns = ['MAKE','FUELTYPE','ENGINESIZE','VEHICLECLASS']
df1
```

Out[122]:

	MAKE	FUELTYPE	ENGINESIZE	VEHICLECLASS
0	ACURA	Z	3.5	SUV-SMALL
1	AUDI	D	3.0	SUV-STANDARD
2	BUICK	E	3.6	MID-SIZE
3	VOLVO	X	2.0	COMPACT

In [123…

```python
size_mapping = {'Z':4,
               'X':3,
               'D':2,
               'E':1}

df1['FUELTYPE'] = df1['FUELTYPE'].map(size_mapping)
df1
```

Out[123]:

	MAKE	FUELTYPE	ENGINESIZE	VEHICLECLASS
0	ACURA	4	3.5	SUV-SMALL
1	AUDI	2	3.0	SUV-STANDARD
2	BUICK	1	3.6	MID-SIZE
3	VOLVO	3	2.0	COMPACT

In [124…

```python
inv_size_mapping = {v:k for k, v in size_mapping.items()}
df1['FUELTYPE'].map(inv_size_mapping)
```

Out[124]:

```
0    Z
1    D
2    E
3    X
Name: FUELTYPE, dtype: object
```

Encoding Class Label

In [125…

```python
import numpy as np
# create a mapping dict
# to convert class labels from strings to integers
class_mapping = {label: idx for idx, label in enumerate(np.unique(df1['VEHICLECLASS']))}
class_mapping
```

Out[125]:

```
{'COMPACT': 0, 'MID-SIZE': 1, 'SUV-SMALL': 2, 'SUV-STANDARD': 3}
```

In [126…

```python
#to convert class labels from strings to integers
df1['VEHICLECLASS'] = df1['VEHICLECLASS'].map(class_mapping)
df1
```

Out[126]:

	MAKE	FUELTYPE	ENGINESIZE	VEHICLECLASS
0	ACURA	4	3.5	2
1	AUDI	2	3.0	3
2	BUICK	1	3.6	1
3	VOLVO	3	2.0	0

In [127…

```
# reverse the class label mapping
inv_class_mapping = {v: k for k, v in class_mapping.items()}
df1['VEHICLECLASS'] = df1['VEHICLECLASS'].map(inv_class_mapping)
df1
```

Out[127]:

	MAKE	FUELTYPE	ENGINESIZE	VEHICLECLASS
0	ACURA	4	3.5	SUV-SMALL
1	AUDI	2	3.0	SUV-STANDARD
2	BUICK	1	3.6	MID-SIZE
3	VOLVO	3	2.0	COMPACT

In [128…

```
from sklearn.preprocessing import LabelEncoder

# Label encoding with sklearn's LabelEncoder
vehicleclass_le = LabelEncoder()
y = vehicleclass_le.fit_transform(df1['VEHICLECLASS'].values)
y
```

Out[128]:

```
array([2, 3, 1, 0])
```

In [129…

```
# reverse mapping
vehicleclass_le.inverse_transform(y)
```

Out[129]:

```
array(['SUV-SMALL', 'SUV-STANDARD', 'MID-SIZE', 'COMPACT'], dtype=object)
```

Performing one-hot encoding on nominal features

In [130…

```
X = df1[['MAKE','FUELTYPE','ENGINESIZE']].values
make_le = LabelEncoder()
X[:, 0] = make_le.fit_transform(X[:,0])
X
```

Out[130]:

```
array([[0, 4, 3.5],
       [1, 2, 3.0],
       [2, 1, 3.6],
       [3, 3, 2.0]], dtype=object)
```

In [131…

```
from sklearn.preprocessing import OneHotEncoder

X = df1[['MAKE','FUELTYPE','ENGINESIZE']].values
make_ohe = OneHotEncoder()
make_ohe.fit_transform(X[:, 0].reshape(-1, 1)).toarray()
```

Out[131]:

```
array([[1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])
```

In [132…

```
from sklearn.compose import ColumnTransformer

X = df1[['MAKE','FUELTYPE','ENGINESIZE']].values
c_transf = ColumnTransformer([('onehot', OneHotEncoder(), [0]),
                              ('nothing', 'passthrough', [1, 2])])
c_transf.fit_transform(X).astype(float)
```

Out[132]:

```
array([[1. , 0. , 0. , 0. , 4. , 3.5],
       [0. , 1. , 0. , 0. , 2. , 3. ],
       [0. , 0. , 1. , 0. , 1. , 3.6],
       [0. , 0. , 0. , 1. , 3. , 2. ]])
```

In [133…

```
# one-hot encoding via pandas
pd.get_dummies(df1[['ENGINESIZE','MAKE','FUELTYPE']])
```

Out[133]:

	ENGINESIZE	FUELTYPE	MAKE_ACURA	MAKE_AUDI	MAKE_BUICK	MAKE_VOLVO
0	3.5	4	1	0	0	0
1	3.0	2	0	1	0	0
2	3.6	1	0	0	1	0
3	2.0	3	0	0	0	1

In [134…

```
#multicollinearity guard in get_dummies
```

```
pd.get_dummies(df1[['ENGINESIZE','MAKE','FUELTYPE']])
```

Out[134]:

	ENGINESIZE	FUELTYPE	MAKE_ACURA	MAKE_AUDI	MAKE_BUICK	MAKE_VOLVO
0	3.5	4	1	0	0	0
1	3.0	2	0	1	0	0
2	3.6	1	0	0	1	0
3	2.0	3	0	0	0	1

In [135…

```
# multicollinearity guard for the OneHotEncoder
make_ohe = OneHotEncoder(categories='auto', drop='first')
c_transf = ColumnTransformer([('onehot', make_ohe, [0]),
                              ('nothing','passthrough', [1, 2])])
c_transf.fit_transform(X).astype(float)
```

Out[135]:

```
array([[0. , 0. , 0. , 4. , 3.5],
       [1. , 0. , 0. , 2. , 3. ],
       [0. , 1. , 0. , 1. , 3.6],
       [0. , 0. , 1. , 3. , 2. ]])
```

Partitioning Dataset in Training and Test Sets

In [136…

```
df2_fuel =pd.read_csv(r"D:\SEMESTER 4\IF540 Machine Learning\LAB\week2\FuelConsumption.csv")

df2_fuel.columns = ['Class label','MODELYEAR','MAKE','MODEL','VEHICLECLASS','ENGINESIZE','CYLINDERS','TRANSMISS
                   'FUELCONSUMPTION_CITY','FUELCONSUMPTION_HWY','FUELCONSUMPTION_COMB','FUELCONSUMPTION_COMB_MP
                   'CO2EMISSIONS']

print('Class labels',np.unique(df2_fuel['Class label']))
df2_fuel.head(5)
```

```
Class labels [   0    1    2 ... 1064 1065 1066]
```

Out[136]:

	Class label	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELCONSUMPTION_CITY
0	0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9
1	1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2
2	2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0
3	3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7
4	4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1

In [137…

```
from sklearn.model_selection import train_test_split
X, y = df2_fuel.iloc[:, 1:].values, df2_fuel.iloc[:, 0].values

X_train, X_test, y_train, y_rest =\
    train_test_split(X, y,
                     test_size=0.3,
                     random_state=0)
#stratitfy di kasus saya ngebuat jadi error, makanya saya hilangkan
```

Bringing Features Onto the Same Scale

In [138…

```
from sklearn.preprocessing import MaxAbsScaler

mms = MaxAbsScaler()
X_train_norm = mms.fit_transform(X_train)
X_test_norm = mms.transform(X_test)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [138], in <cell line: 4>()
      1 from sklearn.preprocessing import MaxAbsScaler
      3 mms = MaxAbsScaler()
----> 4 X_train_norm = mms.fit_transform(X_train)
      5 X_test_norm = mms.transform(X_test)

File ~\anaconda3\lib\site-packages\sklearn\base.py:852, in TransformerMixin.fit_transform(self, X, y, **fit_par
ams)
    848 # non-optimized default implementation; override when a better
    849 # method is possible for a given clustering algorithm
    850 if y is None:
    851     # fit method of arity 1 (unsupervised transformation)
--> 852     return self.fit(X, **fit_params).transform(X)
    853 else:
    854     # fit method of arity 2 (supervised transformation)
    855     return self.fit(X, y, **fit_params).transform(X)

File ~\anaconda3\lib\site-packages\sklearn\preprocessing\_data.py:1150, in MaxAbsScaler.fit(self, X, y)
   1148 # Reset internal state before fitting
   1149 self._reset()
-> 1150 return self.partial_fit(X, y)

File ~\anaconda3\lib\site-packages\sklearn\preprocessing\_data.py:1174, in MaxAbsScaler.partial_fit(self, X, y)
   1153 """Online computation of max absolute value of X for later scaling.
   1154
   1155 All of X is processed as a single batch. This is intended for cases
   (...)
   1171     Fitted scaler.
   1172 """
   1173 first_pass = not hasattr(self, "n_samples_seen_")
-> 1174 X = self._validate_data(
   1175     X,
   1176     reset=first_pass,
   1177     accept_sparse=("csr", "csc"),
   1178     estimator=self,
   1179     dtype=FLOAT_DTYPES,
   1180     force_all_finite="allow-nan",
   1181 )
   1183 if sparse.issparse(X):
   1184     mins, maxs = min_max_axis(X, axis=0, ignore_nan=True)

File ~\anaconda3\lib\site-packages\sklearn\base.py:566, in BaseEstimator._validate_data(self, X, y, reset, vali
date_separately, **check_params)
    564     raise ValueError("Validation should be done on X, y or both.")
    565 elif not no_val_X and no_val_y:
--> 566     X = check_array(X, **check_params)
    567     out = X
    568 elif no_val_X and not no_val_y:

File ~\anaconda3\lib\site-packages\sklearn\utils\validation.py:746, in check_array(array, accept_sparse, accept
_large_sparse, dtype, order, copy, force_all_finite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_featur
es, estimator)
    744         array = array.astype(dtype, casting="unsafe", copy=False)
    745     else:
--> 746         array = np.asarray(array, order=order, dtype=dtype)
    747 except ComplexWarning as complex_warning:
    748     raise ValueError(
    749         "Complex data not supported\n{}\n".format(array)
    750     ) from complex_warning

ValueError: could not convert string to float: 'JEEP'
```

In [139…

```
from sklearn.preprocessing import StandardScaler

stdsc = StandardScaler()
X_train_std = stdsc.fit_transform(X_train)
X_test_std = stdsc.transform(X_test)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [139], in <cell line: 4>()
      1 from sklearn.preprocessing import StandardScaler
      3 stdsc = StandardScaler()
----> 4 X_train_std = stdsc.fit_transform(X_train)
      5 X_test_std = stdsc.transform(X_test)

File ~\anaconda3\lib\site-packages\sklearn\base.py:852, in TransformerMixin.fit_transform(self, X, y, **fit_par
ams)
    848 # non-optimized default implementation; override when a better
    849 # method is possible for a given clustering algorithm
    850 if y is None:
    851     # fit method of arity 1 (unsupervised transformation)
--> 852     return self.fit(X, **fit_params).transform(X)
    853 else:
    854     # fit method of arity 2 (supervised transformation)
    855     return self.fit(X, y, **fit_params).transform(X)

File ~\anaconda3\lib\site-packages\sklearn\preprocessing\_data.py:806, in StandardScaler.fit(self, X, y, sample
_weight)
    804 # Reset internal state before fitting
    805 self._reset()
--> 806 return self.partial_fit(X, y, sample_weight)

File ~\anaconda3\lib\site-packages\sklearn\preprocessing\_data.py:841, in StandardScaler.partial_fit(self, X, y
, sample_weight)
    809 """Online computation of mean and std on X for later scaling.
    810 
    811 All of X is processed as a single batch. This is intended for cases
   (...)
    838     Fitted scaler.
    839 """
    840 first_call = not hasattr(self, "n_samples_seen_")
--> 841 X = self._validate_data(
    842     X,
    843     accept_sparse=("csr", "csc"),
    844     estimator=self,
    845     dtype=FLOAT_DTYPES,
    846     force_all_finite="allow-nan",
    847     reset=first_call,
    848 )
    849 n_features = X.shape[1]
    851 if sample_weight is not None:

File ~\anaconda3\lib\site-packages\sklearn\base.py:566, in BaseEstimator._validate_data(self, X, y, reset, vali
date_separately, **check_params)
    564     raise ValueError("Validation should be done on X, y or both.")
    565 elif not no_val_X and no_val_y:
--> 566     X = check_array(X, **check_params)
    567     out = X
    568 elif no_val_X and not no_val_y:

File ~\anaconda3\lib\site-packages\sklearn\utils\validation.py:746, in check_array(array, accept_sparse, accept
_large_sparse, dtype, order, copy, force_all_finite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_featur
es, estimator)
    744         array = array.astype(dtype, casting="unsafe", copy=False)
    745     else:
--> 746         array = np.asarray(array, order=order, dtype=dtype)
    747 except ComplexWarning as complex_warning:
    748     raise ValueError(
    749         "Complex data not supported\n{}\n".format(array)
    750     ) from complex_warning

ValueError: could not convert string to float: 'JEEP'
```

In [140…

```
ex = np.array([0, 1 , 2, 3 ,4, 5])

print('standardized:', (ex - ex.mean()) / ex.std())

#normalize
print('normalized: ', (ex - ex.min()) / (ex.max() - ex.min()))
```

```
standardized: [-1.46385011 -0.87831007 -0.29277002  0.29277002  0.87831007  1.46385011]
normalized:  [0.  0.2 0.4 0.6 0.8 1. ]
```

Kesimpulan

Berikan simpulan yang dilakukan dari hasil kerja menggunakan algoritma dan 2 dataset yang dipilih. Simpulan bisa berkisar antara (bisa di modifikasi):

```
- kesimpulan dari lab minggu ini adalah bagaimana kita menggunakan
machine learning dalam memproses sebuah data, dengan teknik train test
,dan data preprocessing. hal ini penting supaya kita bisa mendapatkan
hasil dari setiap test yang sudah dilakukan contohnya pada dataset
```

```
vaccine dengan dataset fuel consumption
```

In [141…

```python
# Footer
myDate = datetime.datetime.now()
print("I certify that this is my own work.")
print("Signed by:")
print("Name: \t\t{}".format(myName))
print("NIM: \t\t{}".format(myNIM))
print("Time-stamp:\t{}".format(myDate))
```

```
I certify that this is my own work.
Signed by:
Name:           Christopher Darren
NIM:            00000054804
Time-stamp:     2023-02-17 19:38:34.514598
```

Save the notebook, then convert the notebook to html (by running the next code).

In [ ]:

```
!jupyter nbconvert --to html "./IF540_Kelas EL_00000054804_Christopher Darren_Week02.ipynb" --output-dir="./"
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js