Experiment - 02

Aim: To apply Data Cleaning techniques

Theory: Data Cleaning using Pandas in Python is the most important task that a data science professional should do. Wrong or bad-quality data can be detrimental to processes and analysis. Clean data will ultimately increase overall productivity and permit the very best quality information in decision-making.

Data Cleaning Cycle: It is the method of analyzing, distinguishing, and correcting untidy, raw data. Python Pandas Data Cleaning involves filling in missing values, handling outliers, and distinguishing and fixing errors in the dataset. Meanwhile, the techniques used for data cleaning in data science using Python might vary in step with different types of datasets. In this tutorial, we will learn how to clean data using pandas.

Signs of an untidy dataset

We have to take a closer look to find common signs of a messy dataset. These common signs are as follows:-

- **Missing numerical data:** Missing numerical data needs to be identified and addressed. Either they need to be deleted or replaced with a suitable test statistic.
- **Untidy data:** Untidy dataset can contain multiple problems. They prevent us from transforming the messy dataset into a clean dataset that is suitable for analysis.
- **Unexpected data values:** Mismatched data types of a column and data values can cause potential problems. They need to be investigated and solved.
- **Inconsistent column names:** Column names contain inconsistent capitalizations and bad characters. They need to be addressed properly.
- **Outliers:** Outliers need to be detected. They pose potential problems that need to be investigated and removed.
- **Duplicate rows and columns:** Duplicate rows and columns make data redundant. They can bias an analysis. Hence, they need to be found and dropped.

Code:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris

data = "/content/sample_data/fictitious_dataset.csv"
df = pd.read_csv(data)

print(df.shape)
print(df.head())
print(df.tail())
print(df.info())
print(df.dtypes)
print(df.describe)
print(df.columns)
```

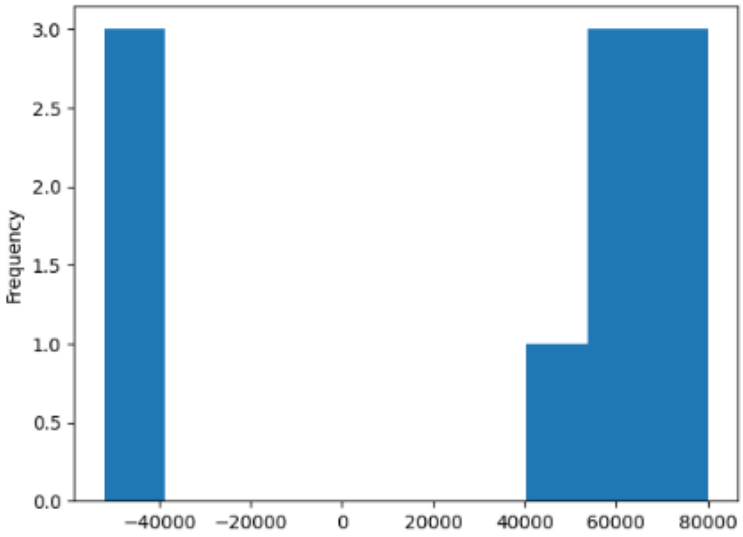
```
(10, 9)
   id age_sex  height_cm  weight_kg  income_usd   bmi  hours_sleep  exercise_hours_weekly city
0   1   25_M      175         70         50000    23.5       6                5.0            NY
1   2   34_F      160         55        -45000    21.1       7                NaN            LA
2   3   29_M      180         82         60000    25.3       5                4.0            TX
3   4   42_F      158         50         55000    20.1       8                6.0            SF
4   5   31_M      172         76         70000    26.1       6                NaN           CHI
   id age_sex  height_cm  weight_kg  income_usd   bmi  hours_sleep  exercise_hours_weekly city
5   6   27_F      165         59        -52000    21.7       7                3.0           MIA
6   7   38_M      178         88         80000    28.4       4                7.0           SEA
7   8   50_F      155         48         62000    19.9       9                2.0           DEN
8   9   22_M      182         85         73000    27.0       3                8.0           HOU
9  10   45_F      159         52        -49000    20.5      10                NaN           BOS
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 9 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    10 non-null     int64
 1   age_sex               10 non-null     object
 2   height_cm             10 non-null     int64
 3   weight_kg             10 non-null     int64
 4   income_usd            10 non-null     int64
 5   bmi                   10 non-null     float64
 6   hours_sleep           10 non-null     int64
 7   exercise_hours_weekly  7 non-null     float64
 8   city                  10 non-null     object
dtypes: float64(2), int64(5), object(2)
memory usage: 852.0+ bytes
```

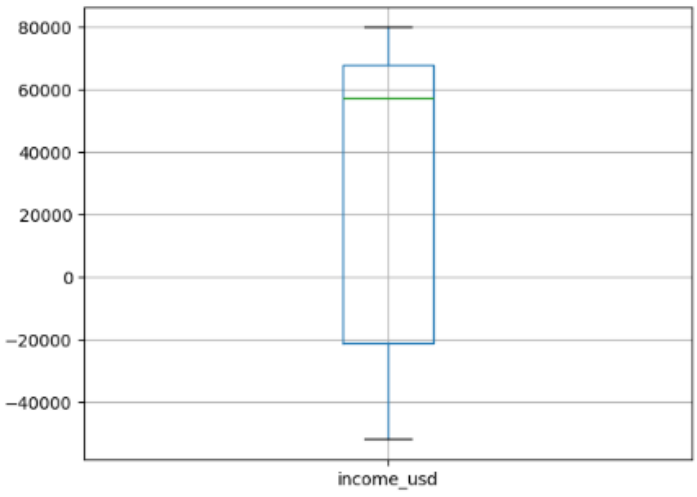
```
id                         int64
age_sex                   object
height_cm                  int64
weight_kg                  int64
income_usd                 int64
bmi                      float64
hours_sleep                int64
exercise_hours_weekly    float64
city                      object
dtype: object
```

```
<bound method NDFrame.describe of    id age_sex  height_cm  weight_kg  income_usd   bmi  hours_sleep  exercise_hours_weekly city
0   1   25_M      175         70         50000    23.5       6                5.0           NY
1   2   34_F      160         55        -45000    21.1       7                NaN           LA
2   3   29_M      180         82         60000    25.3       5                4.0           TX
3   4   42_F      158         50         55000    20.1       8                6.0           SF
4   5   31_M      172         76         70000    26.1       6                NaN          CHI
5   6   27_F      165         59        -52000    21.7       7                3.0          MIA
6   7   38_M      178         88         80000    28.4       4                7.0          SEA
7   8   50_F      155         48         62000    19.9       9                2.0          DEN
8   9   22_M      182         85         73000    27.0       3                8.0          HOU
9  10   45_F      159         52        -49000    20.5      10                NaN          BOS>
Index(['id', 'age_sex', 'height_cm', 'weight_kg', 'income_usd', 'bmi', 'hours_sleep',
       'exercise_hours_weekly', 'city'],
      dtype='object')
```

```
df['income_usd'].plot(kind='hist')
plt.show()
```




```
df.boxplot(column='income_usd')
plt.show()
```



```
df[['age','sex']] = df.age_sex.str.split("_", expand = True)
df.drop(['age_sex'], axis=1, inplace=True)
df = df[['id', 'age', 'sex', 'height_cm', 'weight_kg', 'income_usd',
'bmi', 'hours_sleep',
       'exercise_hours_weekly', 'city']]

df['income_usd'].replace(-45000, 45000, inplace=True)
df['income_usd'].replace(-52000, 52000, inplace=True)
df['income_usd'].replace(-49000, 49000, inplace=True)

df.isnull().sum()
mean = df['exercise_hours_weekly'].mean()
df['exercise_hours_weekly'].fillna(value = mean, inplace=True)
df
```

	id	age	sex	height_cm	weight_kg	income_usd	bmi	hours_sleep	exercise_hours_weekly	city
0	1	25	M	175	70	50000	23.5	6	5.0	NY
1	2	34	F	160	55	45000	21.1	7	5.0	LA
2	3	29	M	180	82	60000	25.3	5	4.0	TX
3	4	42	F	158	50	55000	20.1	8	6.0	SF
4	5	31	M	172	76	70000	26.1	6	5.0	CHI
5	6	27	F	165	59	52000	21.7	7	3.0	MIA
6	7	38	M	178	88	80000	28.4	4	7.0	SEA
7	8	50	F	155	48	62000	19.9	9	2.0	DEN
8	9	22	M	182	85	73000	27.0	3	8.0	HOU
9	10	45	F	159	52	49000	20.5	10	5.0	BOS

Conclusion: Hence, we performed data cleaning on a fictitious dataset using Pandas