PRACTICAL :2 Data cleaning and handling the missing values in ML

# Sample data for demonstration purposes

data = {

'Name': ['John', 'Jane', 'Mary', None, 'Tom', 'John'],

'Age': [28, 22, 35, 29, None, 28],

'Email': ['john@example.com', 'jane@example.com', None, 'mary@example.com', 'tom@example.com', 'john@example.com']

}

# Create a DataFrame

df = pd.DataFrame(data)

# Display original DataFrame

print("Original DataFrame:")

print(df)

# 1. Handle missing values

# Option 1: Drop rows with any missing values

df\_cleaned = df.dropna()

# Option 2: Fill missing values (e.g., fill with a specific value or forward fill)

# df\_cleaned = df.fillna({'Name': 'Unknown', 'Age': df['Age'].mean(), 'Email': 'noemail@example.com'})

# 2. Remove duplicate rows

df\_cleaned = df\_cleaned.drop\_duplicates()

# 3. Standardize column names (e.g., make them all lower case)

df\_cleaned.columns = [col.lower() for col in df\_cleaned.columns]

df["Email"].fillna('padma@example.com',inplace=True)

# Display cleaned DataFrame

print("\nCleaned DataFrame:")

print(df)

print(df\_cleaned)

output:

Original DataFrame:

Name Age Email

0 John 28.0 john@example.com

1 Jane 22.0 jane@example.com

2 Mary 35.0 None

3 None 29.0 mary@example.com

4 Tom NaN tom@example.com

5 John 28.0 john@example.com

Cleaned DataFrame:

Name Age Email

0 John 28.0 john@example.com

1 Jane 22.0 jane@example.com

2 Mary 35.0 padma@example.com

3 None 29.0 mary@example.com

4 Tom NaN tom@example.com

5 John 28.0 john@example.com

name age email

0 John 28.0 john@example.com

1. Jane 22.0 [jane@example.com](mailto:jane@example.com)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**OUTLIER DETECTION:**

* The dataset used in this article is the Diabetes dataset and it is preloaded in the Sklearn library.
* Outliers can be detected using visualization, implementing mathematical formulas on the dataset, or using the statistical approach.

import sklearn

from sklearn.datasets import load\_diabetes

import pandas as pd

import matplotlib.pyplot as plt

# Load the dataset

diabetics = load\_diabetes()

# Create the dataframe

column\_name = diabetics.feature\_names

df\_diabetics = pd.DataFrame(diabetics.data)

df\_diabetics.columns = column\_name

print(df\_diabetics.head())

**Visualizing and Removing Outliers Using Box Plot**

Scikit:

* Simple and efficient tools for predictive data analysis
* Accessible to everybody, and reusable in various contexts
* Built on NumPy, SciPy, and matplotlib
* Open source, commercially usable - BSD license

BOX PLOT:

Prerequisite: seaborn

* The Boxplots are used to visualize the distribution of data which is useful when a comparison of data is required.
* Sometimes, Boxplot is also known as a box-and-whisker plot. The box shows the quartiles of dataset and whiskers extend to show rest of the distribution.

**Horizontal Box plots**

Seaborn uses the boxplot() method to draw a boxplot.

We can turn the boxplot into a horizontal boxplot by two methods first, we need to switch x and y attributes and pass it to the boxplot( ) method, and the other is to use the orient=”h” option and pass it to the boxplot() method.

# import library & dataset

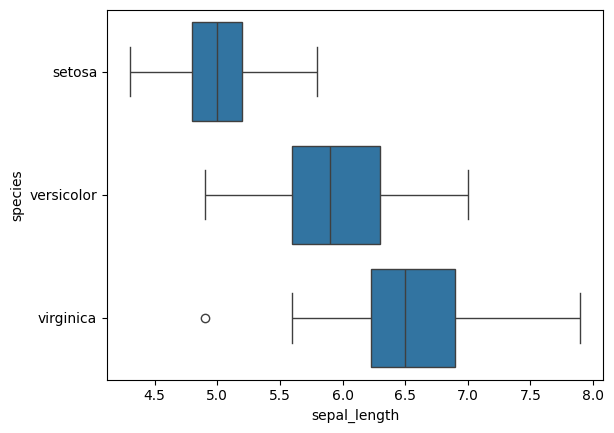
import seaborn as sns

df = sns.load\_dataset('iris')

# Just switch x and y

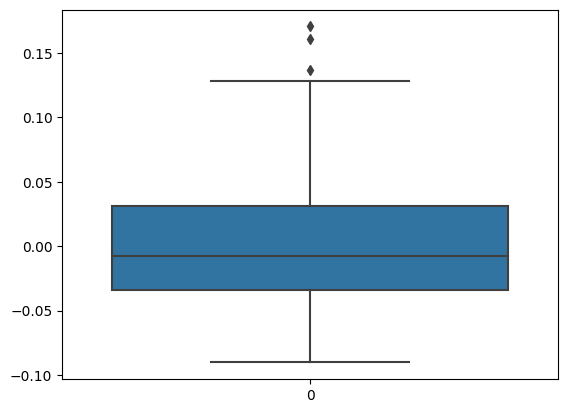
sns.boxplot(y=df["species"], x=df["sepal\_length"])

output:



import seaborn as sns

sns.boxplot(df\_diabetics['bmi'])



import seaborn as sns

import matplotlib.pyplot as plt

def removal\_box\_plot(df, column, threshold):

sns.boxplot(df[column])

plt.title(f'Original Box Plot of {column}')

plt.show()

removed\_outliers = df[df[column] <= threshold]

sns.boxplot(removed\_outliers[column])

plt.title(f'Box Plot without Outliers of {column}')

plt.show()

return removed\_outliers

threshold\_value = 0.12

no\_outliers = removal\_box\_plot(df\_diabetics, 'bmi', threshold\_value)

