Importing files

# Importing the libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

# Ignore harmless warnings

import warnings

warnings.filterwarnings("ignore")

# Set to display all columns in dataset

pd.set\_option("display.max\_columns", 50)

# import psql to run queries

import pandasql as psql

Load the datasets

1) bmidata = pd.read\_csv(r”copy\_path”, header = 0) – CSV file

<https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html>

2) bmidata = pd. read\_excel (r”copy\_path”, header = 0) ---- Excel file

<https://pandas.pydata.org/docs/reference/api/pandas.read_excel.html>

If more than 1 excel sheet is present in the excel file

pd.read\_excel(r” copy\_path ”, sheet\_name=0)

To Load the data from all the work sheets

Pd.concat(pd.read\_excel(r”copy\_path”, sheet\_name=None, ignore\_index = True)

Exploratory Data Analysis

1 ) # Identify the number of features or columns

len(bmidata.columns)

2) # Identify the features

bmidata.columns

3) # Identify the size of the dataset

bmidata.shape

4) # Identify the datatypes of the features or columns

bmidata.dtypes

5) # Display the datatypes of the features or columns / column index / Non-null count / Range

bmidata.info()

6) # Delecting the 8 columns

# del CO2Emission['year']

CO2EmissionN = CO2Emission.drop([‘year’], axis = 1)

7) Removing Duplicate values

# Display Dupicate values with in dataset

BMCdata\_dup = BMCdata[BMCdata.duplicated(keep='last')]

BMCdata\_dup

# Remove the identified duplicate records

BMCdata = BMCdata.drop\_duplicates()

BMCdata.shape

# reset the index

BMCdata = BMCdata.reset\_index(drop=True)

**Data Wrangling or Missing values**

1. # Checking if the dataset has empty cells

bmidata.isnull().values.any()

1. # Identify the number of empty cells

bmidata.isnull().sum()

Identify the Target and Independent Variable

1. # Identify the independent variable

x = pd.DataFrame(bmidata1['Weight kg'])

# Identify the independent variable

y = pd.DataFrame(bmidata1['BMI'])

1. # Identify the independent and Depentent (target) variables

x1 = bmidata.iloc[:,0:3]

y1 = bmidata.iloc[:, 3:4]

# Identify the independent and Depentent (target) variables

x2 = bmidata.iloc[:,:-1] # independent variables

y2 = bmidata.iloc[:,-1] # dependent (target) variable

1. # Identify the Target and Independent Variables

IndepVar =[]

for col in CO2EmissionN.columns:

if col != 'CO2\_Emissions':

IndepVar.append(col)

TargetVar = 'CO2\_Emissions'

x = CO2EmissionN[IndepVar]

y = CO2EmissionN[TargetVar]

Feature Importance

1. # Get feature importance

from matplotlib import pyplot

importance = dcreg.feature\_importances\_

# Summarize feature importance

for i,v in enumerate(importance):

print('Feature: %0d, Score: %.5f' % (i,v))

# Plot feature importance

pyplot.bar([x for x in range(len(importance))], importance)

pyplot.show()

**Tree Diagram in Jupitor notebook**

1)

from sklearn import tree

tree.plot\_tree(BHdataDR)

1. **To save png file in folder**

fn=['V1','V2','V3','V4', 'V5', 'V6', 'V7', 'V8','V9','V10','V11','V12','V13']

cn=['1', '2', '3', '4', '5']

fig, axes = plt.subplots(nrows = 1,ncols = 1,figsize = (4,4), dpi=300)

tree.plot\_tree(BHdataDR, feature\_names = fn, class\_names=cn, filled = True);

fig.savefig('BHdataDR.png')

**Random Forest Diagram**

# Visualize individual trees and code below visualizes the first decision tree of Random Forest

from sklearn import tree

fn1 = cols # All independent features

cn1 =['0','1']

fig, axes = plt.subplots(nrows = 1, ncols=1, figsize=(4,4), dpi=800) # dpi means depth

tree.plot\_tree(loansRF.estimators\_[0], feature\_names=fn1, class\_names=cn1, filled = True);

fig.savefig('loansRF1.png')

2)

# Visualize individual trees and code below visualizes the first 5 decision trees of Random Forest

from sklearn import tree

fn2=cols

cn2=['0', '1']

fig, axes = plt.subplots(nrows = 1, ncols = 5, figsize = (10,2), dpi=3000)

for index in range(0, 5):

tree.plot\_tree(loansRF.estimators\_[index], feature\_names = fn2, class\_names=cn2,

filled = True,

ax = axes[index]);

axes[index].set\_title('Estimator: ' + str(index), fontsize = 11)

fig.savefig('loansRF5.png')