Algonquin College of Applied Arts and Technology

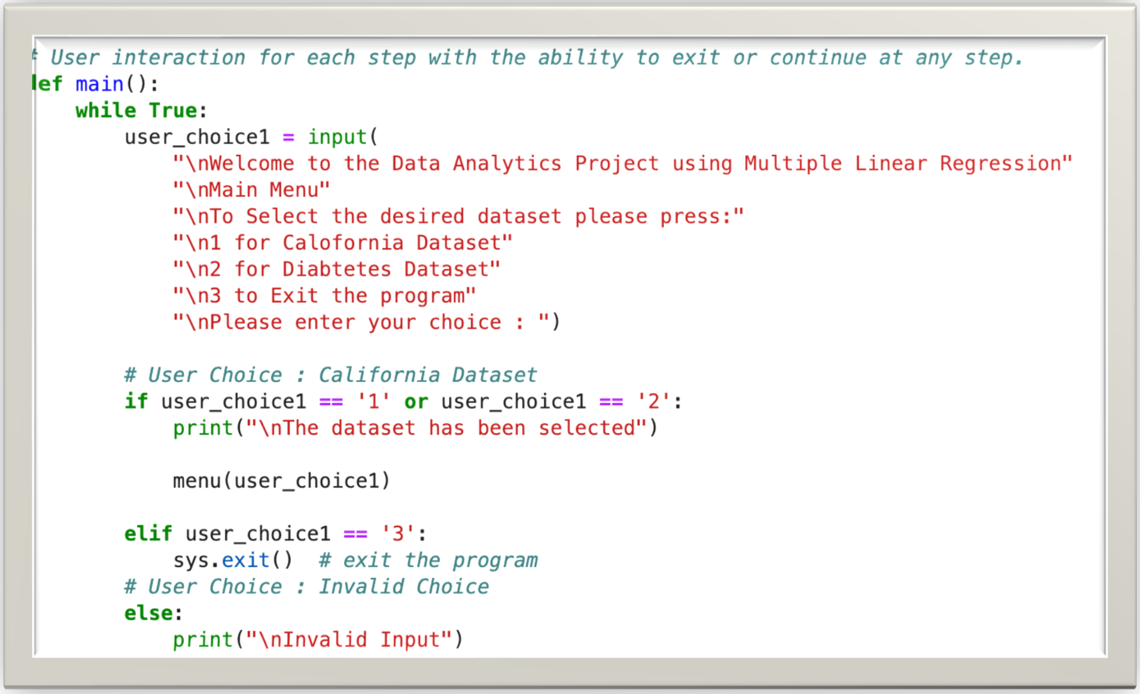
Business Intelligence System Infrastructure

22F_CST2101 – Business Intelligence Programming

Final Project

on

Data Analysis through Multiple Linear Regression using Python



```
# User interaction for each step with the ability to exit or continue at any step.
def main():
    while True:
        user_choice1 = input(
            "\nWelcome to the Data Analytics Project using Multiple Linear Regression"
            "\nMain Menu"
            "\nTo Select the desired dataset please press:"
            "\n1 for Calofornia Dataset"
            "\n2 for Diabtetes Dataset"
            "\n3 to Exit the program"
            "\nPlease enter your choice : ")

        # User Choice : California Dataset
        if user_choice1 == '1' or user_choice1 == '2':
            print("\nThe dataset has been selected")

            menu(user_choice1)

        elif user_choice1 == '3':
            sys.exit()  # exit the program
        # User Choice : Invalid Choice
        else:
            print("\nInvalid Input")
```

Prepared By

First Name	Last Name	Student ID
Wali	Hyder	041057663

Under the kind guidance of Mr. Steve Conrad | Professor, Business Intelligence Programming

Submission Date: 18th December 2022

Table of Contents

1 EXECUTIVE SUMMARY

I am pleased to present a report on the Topic: ***"Data Analysis through Multiple Linear Regression using Python"***. This work comes along with the Python File (.py) file and the relevant test documents prepared by Wali Hyder to fulfill the requirement of the final project for the CST2101 – Business Intelligence Programming.

Through meticulous qualitative & quantitative data analysis and machine learning models, efforts have been made to allow user to choose from the following datasets from sklearn dataset library. The datasets are :
(SKLEARN Dataset Library - Toy Dataset)
(SKLEARN Dataset Library - Real World Dataset)

1. California Housing Dataset (Real-world)
2. Diabetes Dataset (Toy)

The IPython Notebook tool used for writing the code , training, testing and visualization is JupyterLab.

After the selection of the dataset, the following steps are performed in the order:

1. Load the dataset
2. Explore the data
3. Split the data for training and testing
4. Train the data model
5. Test the data model
6. Visualize the expected vs. predicted
7. Create the regression model metrics

Also , effort is made to add features like user interaction allowing the user to continue or exit at any step. The user is also prevented from performing the next steps before the previous step is executed. for e.g. , Training the data model before loading or splitting.

2 DESIGN & PSEUDOCODE

The approach taken to write code is explained in this section. I have defined two menus that will be called initially to allow user interaction. The pseudo code logical diagram along with the steps followed is briefly explained here.

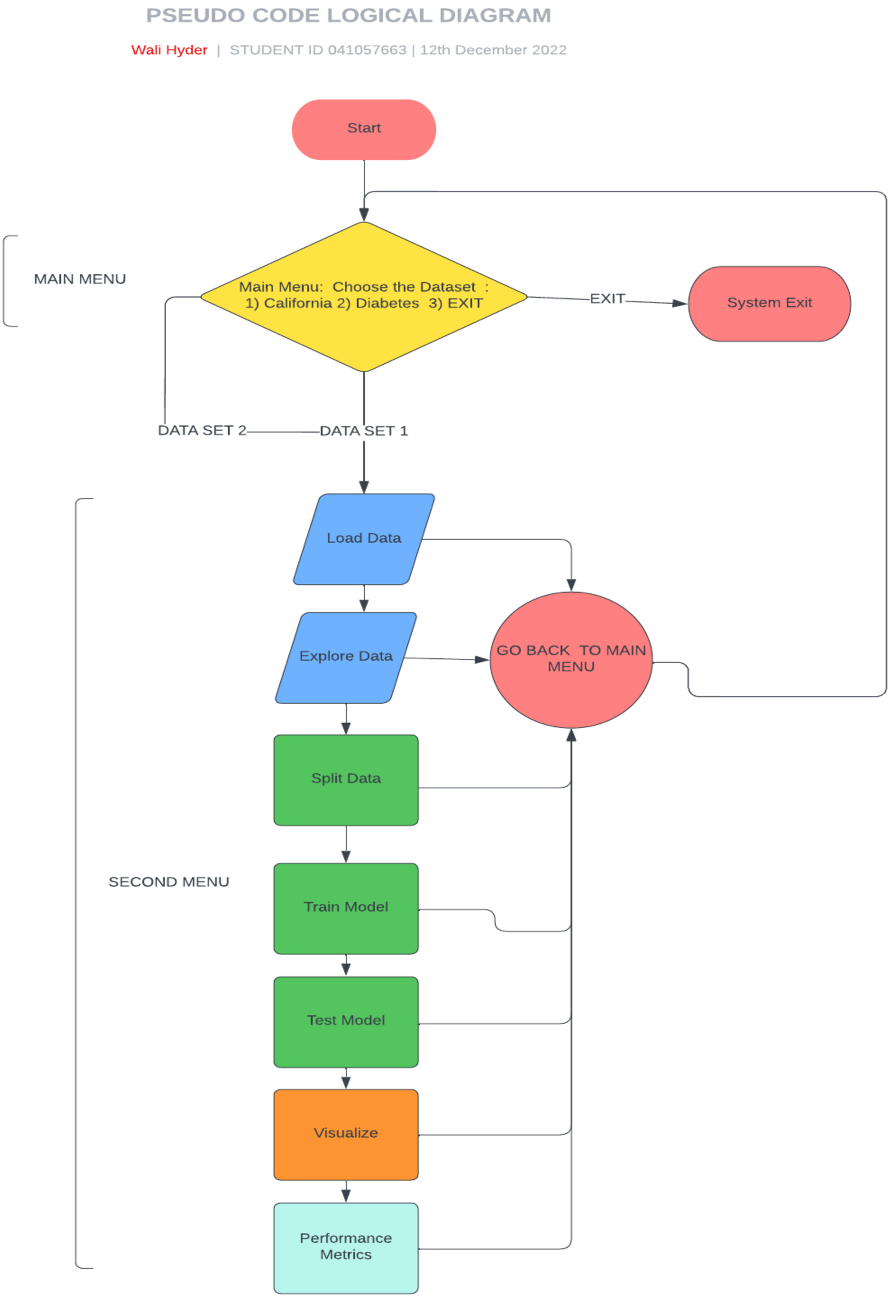


Figure 1 Pseudocode Logical Diagram

Step1 : Importing the required libraries
All the libraries were imported

Step 2 : Two Menus were defined :

I. Main Menu : To give user to choose from the two datasets. Also, user interaction to exit the program at this stage.
II. Second Menu : To allow user to choose from the list of operations. Also, user interaction to go back to the main menu. This menu is called after every step

Step 3: Loading the dataset with prevention to from performing the next steps before the previous step is executed. here I have introduced an exception handling and multiple nested functions to prevent user to skip the steps. I have also assigned the Boolean variables to be used as indicators.

Step 4: Exploratory Data Analysis (EDA) : Function is defined to Explore the data. Guidance has been taken from the book in reference.

Step5 : Splitting the Dataset : function is defined to spilt the data. only after dataset is split into test and train (train- 75% & Test- 25% default from sklearn). Training & Testing can be done.

Step 6 : Training the Model : function is defined to train the model with exception handling loop.

Step 7 Testing the model : function is defined to test the model after training with exception handling loop.

Step 8 Performance Analysis - Regression Metrics : function is defined to evaluate the performance. Mean squared error and coefficient of determination has been computed.

Step 9 Visualization : Function is defined to visualize using linear regression between the expected vs predicted values.

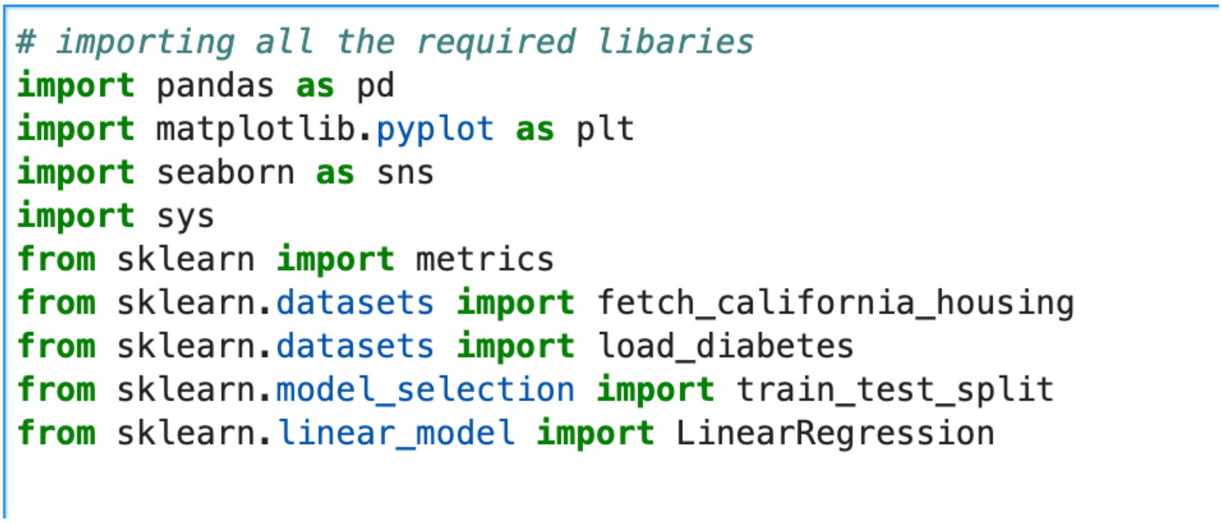
Step 10 : At every step in the second main menu the user had an option to continue or go back to the main menu. Also, user was prevented from performing the next steps from the previous ones.

3 DATA ANALYSIS USING PYTHON – CODE

The analysis of our data will be carried out in line with the problem statement put forward. The following subsections will address the problem statements in detail by providing a supporting analysis from the models and visualizations performed.

THE SNIPPETS FOR THE SECTIONS

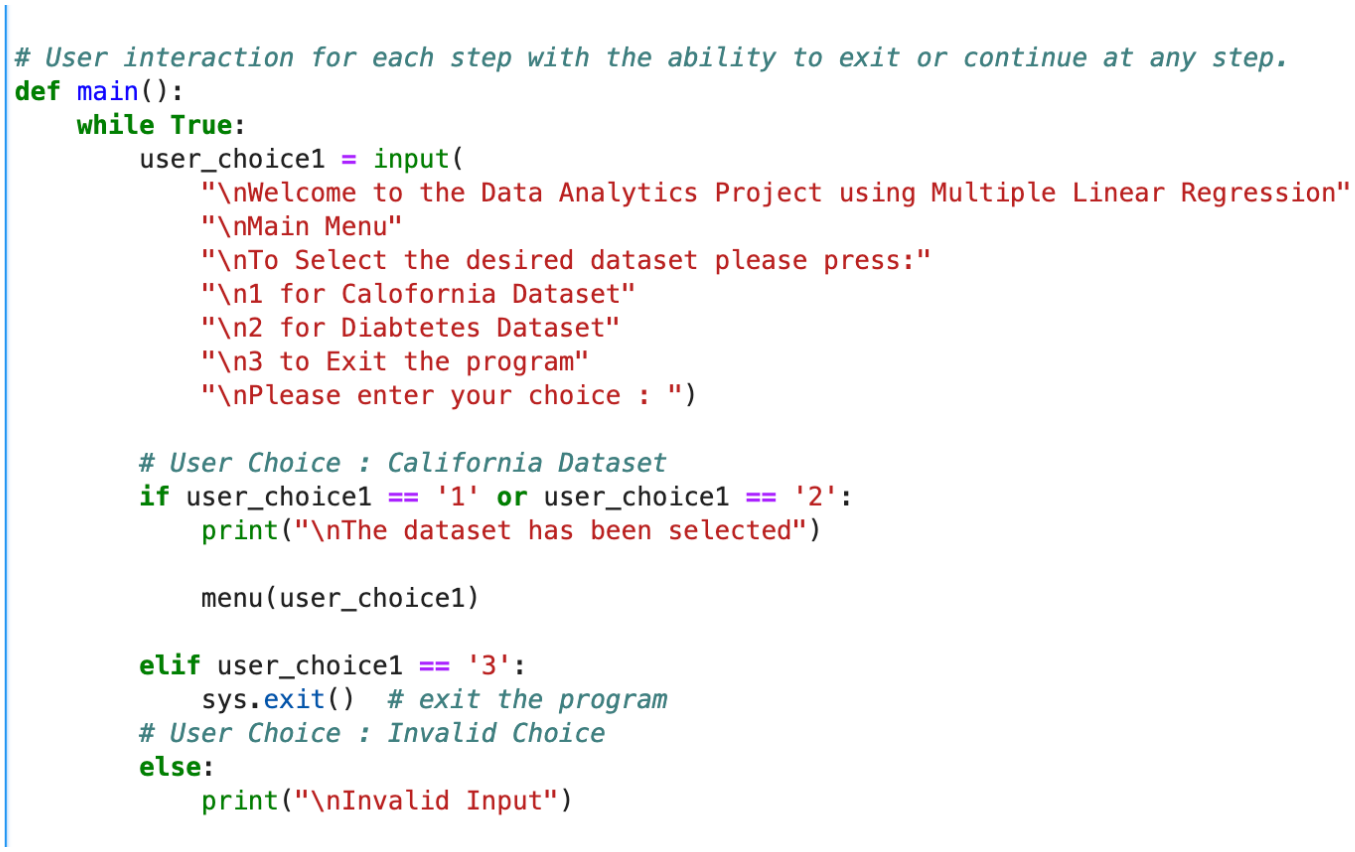
I. Importing the required libraries



```
# importing all the required libaries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sys
from sklearn import metrics
from sklearn.datasets import fetch_california_housing
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

Snippet 1 Libraries Imported

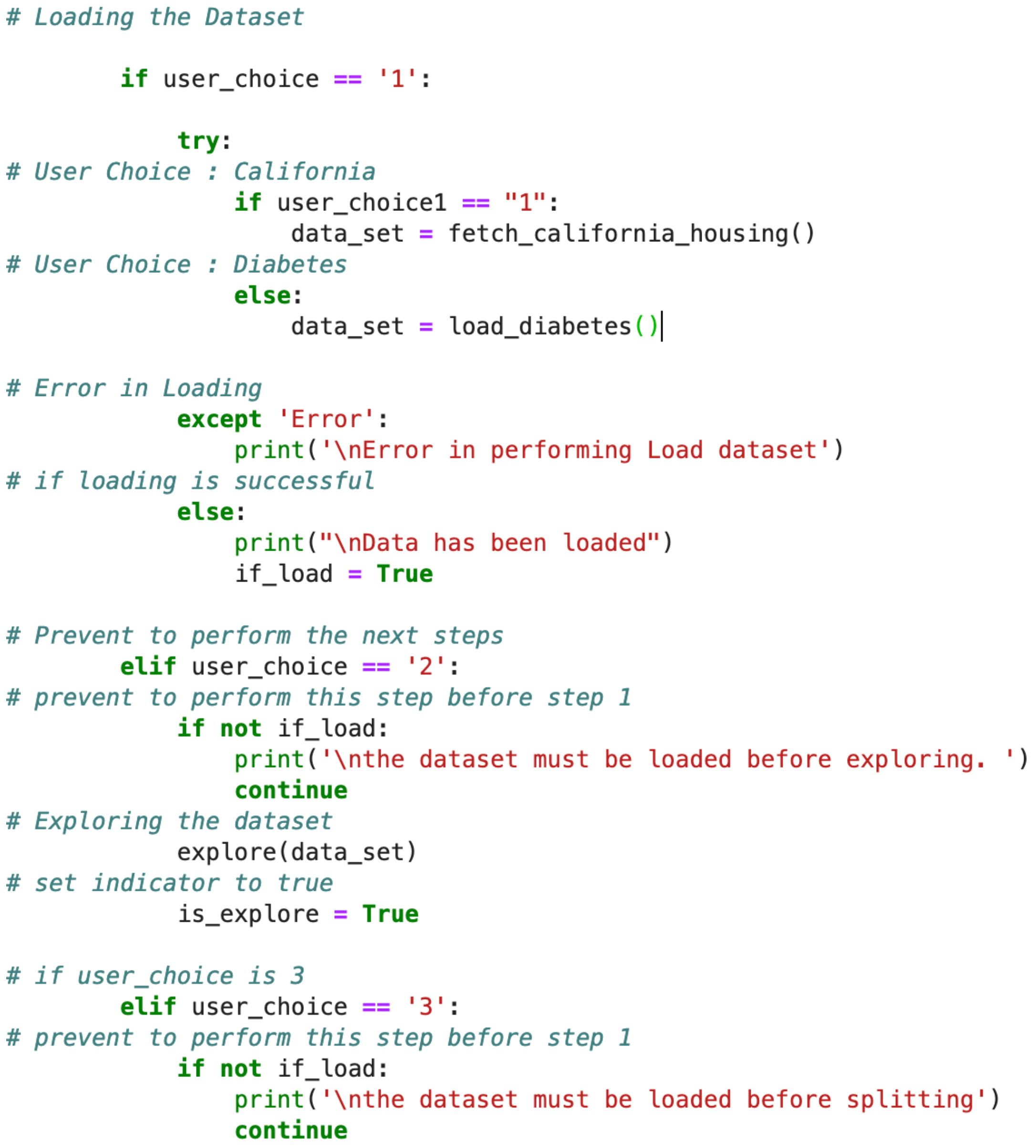
II. User Interaction and dataset selection



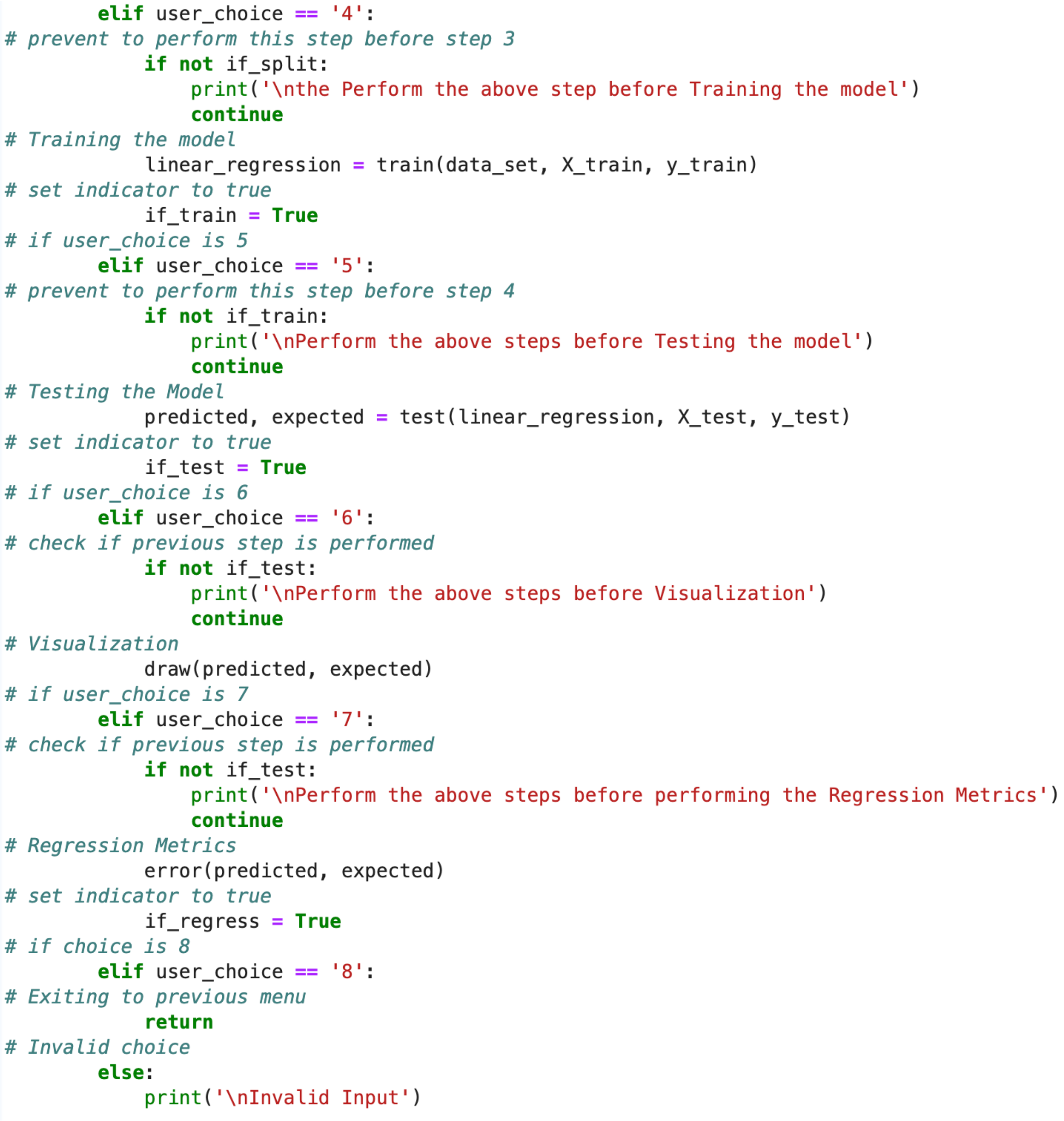
```python
# User interaction for each step with the ability to exit or continue at any step.
def main():
    while True:
        user_choice1 = input(
            "\nWelcome to the Data Analytics Project using Multiple Linear Regression"
            "\nMain Menu"
            "\nTo Select the desired dataset please press:"
            "\n1 for Calofornia Dataset"
            "\n2 for Diabtetes Dataset"
            "\n3 to Exit the program"
            "\nPlease enter your choice : ")

        # User Choice : California Dataset
        if user_choice1 == '1' or user_choice1 == '2':
            print("\nThe dataset has been selected")

            menu(user_choice1)

        elif user_choice1 == '3':
            sys.exit()  # exit the program
        # User Choice : Invalid Choice
        else:
            print("\nInvalid Input")
```

Snippet 2 User interaction - Dataset selection

III. Loading the dataset with prevention to from performing the next steps before the previous step is executed.

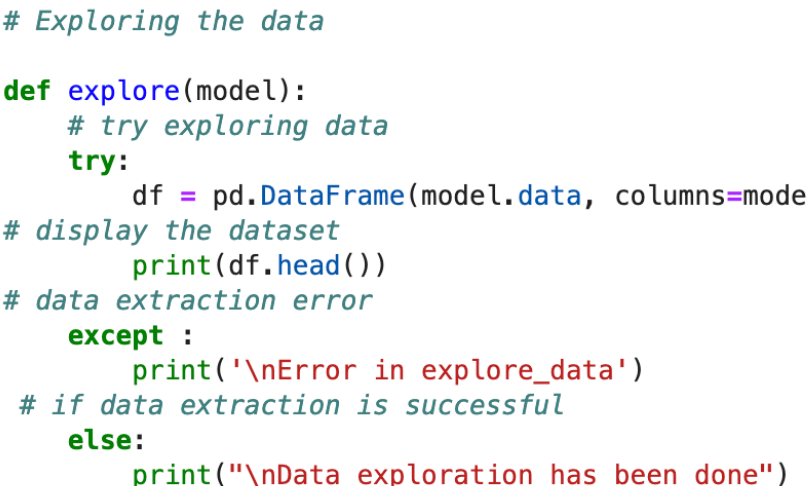


```
# Loading the Dataset

        if user_choice == '1':

            try:
# User Choice : California
                if user_choice1 == "1":
                    data_set = fetch_california_housing()
# User Choice : Diabetes
                else:
                    data_set = load_diabetes()

# Error in Loading
            except 'Error':
                print('\nError in performing Load dataset')
# if loading is successful
            else:
                print("\nData has been loaded")
                if_load = True

# Prevent to perform the next steps
        elif user_choice == '2':
# prevent to perform this step before step 1
            if not if_load:
                print('\nthe dataset must be loaded before exploring. ')
                continue
# Exploring the dataset
            explore(data_set)
# set indicator to true
            is_explore = True

# if user_choice is 3
        elif user_choice == '3':
# prevent to perform this step before step 1
            if not if_load:
                print('\nthe dataset must be loaded before splitting')
                continue
```



```
        elif user_choice == '4':
# prevent to perform this step before step 3
            if not if_split:
                print('\nthe Perform the above step before Training the model')
                continue
# Training the model
            linear_regression = train(data_set, X_train, y_train)
# set indicator to true
            if_train = True
# if user_choice is 5
        elif user_choice == '5':
# prevent to perform this step before step 4
            if not if_train:
                print('\nPerform the above steps before Testing the model')
                continue
# Testing the Model
            predicted, expected = test(linear_regression, X_test, y_test)
# set indicator to true
            if_test = True
# if user_choice is 6
        elif user_choice == '6':
# check if previous step is performed
            if not if_test:
                print('\nPerform the above steps before Visualization')
                continue
# Visualization
            draw(predicted, expected)
# if user_choice is 7
        elif user_choice == '7':
# check if previous step is performed
            if not if_test:
                print('\nPerform the above steps before performing the Regression Metrics')
                continue
# Regression Metrics
            error(predicted, expected)
# set indicator to true
            if_regress = True
# if choice is 8
        elif user_choice == '8':
# Exiting to previous menu
            return
# Invalid choice
        else:
            print('\nInvalid Input')
```

IV. Exploratory Data Analysis (EDA)



```
# Exploring the data

def explore(model):
    # try exploring data
    try:
        df = pd.DataFrame(model.data, columns=model.feature_names)
# display the dataset
        print(df.head())
# data extraction error
    except :
        print('\nError in explore_data')
 # if data extraction is successful
    else:
        print("\nData exploration has been done")
```

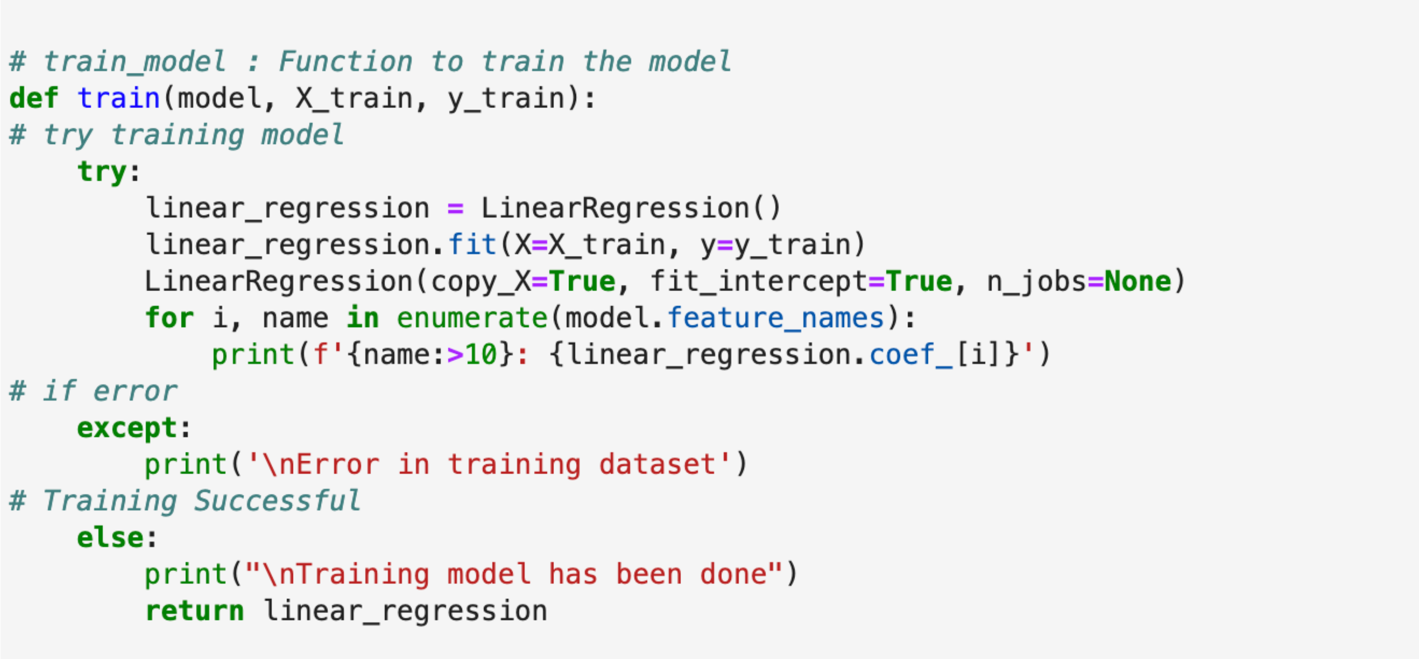
Snippet 3 Exploring the data

V. Splitting the Dataset

```
# split_data : Function to split the dataset
def split(model):
# try spliting data
    try:
        X_train, X_test, y_train, y_test = train_test_split(model.data, model.target, random_state=11)
# if error
    except :
        print("\nError in splitting the data")
# sent successful message
    else:
        print("\nData splitting has been done")
        return X_train, X_test, y_train, y_test
```

Snippet 4 Split data

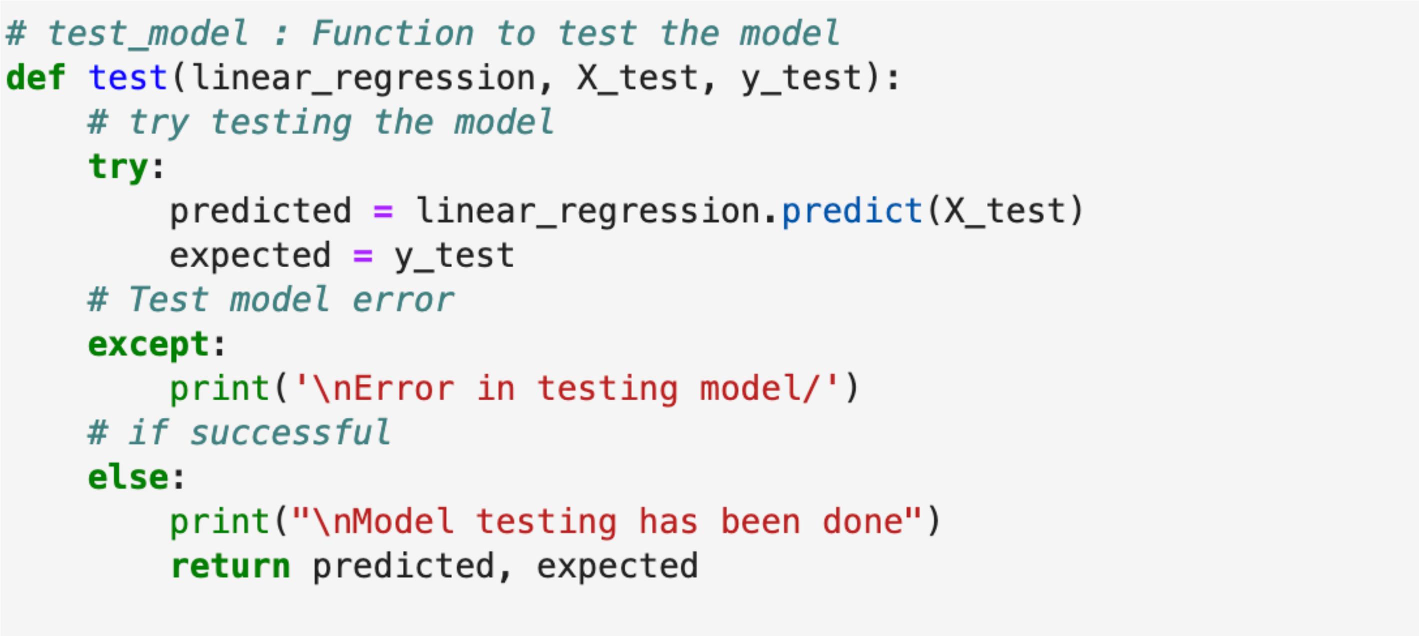
VI. Training the Model



```
# train_model : Function to train the model
def train(model, X_train, y_train):
# try training model
    try:
        linear_regression = LinearRegression()
        linear_regression.fit(X=X_train, y=y_train)
        LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None)
        for i, name in enumerate(model.feature_names):
            print(f'{name:>10}: {linear_regression.coef_[i]}')
# if error
    except:
        print('\nError in training dataset')
# Training Successful
    else:
        print("\nTraining model has been done")
        return linear_regression
```

Snippet 5 Train the model

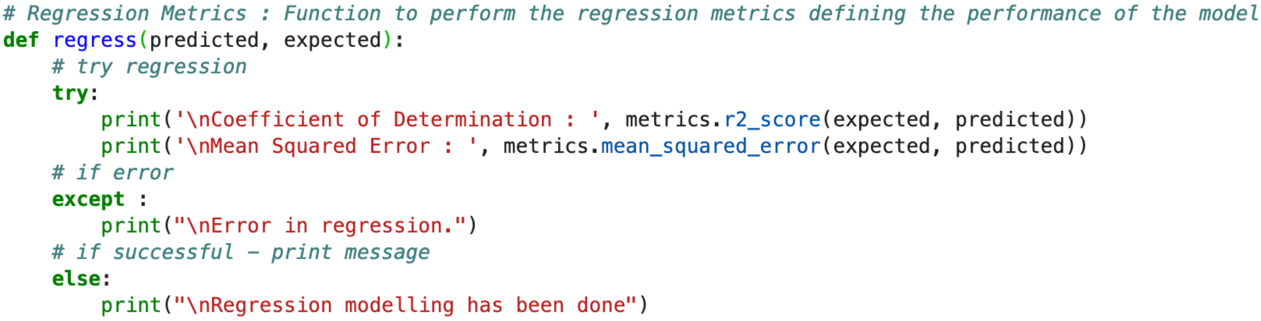
VII. Testing the model



```
# test_model : Function to test the model
def test(linear_regression, X_test, y_test):
    # try testing the model
    try:
        predicted = linear_regression.predict(X_test)
        expected = y_test
    # Test model error
    except:
        print('\nError in testing model/')
    # if successful
    else:
        print("\nModel testing has been done")
        return predicted, expected
```

Snippet 6 Testing the model

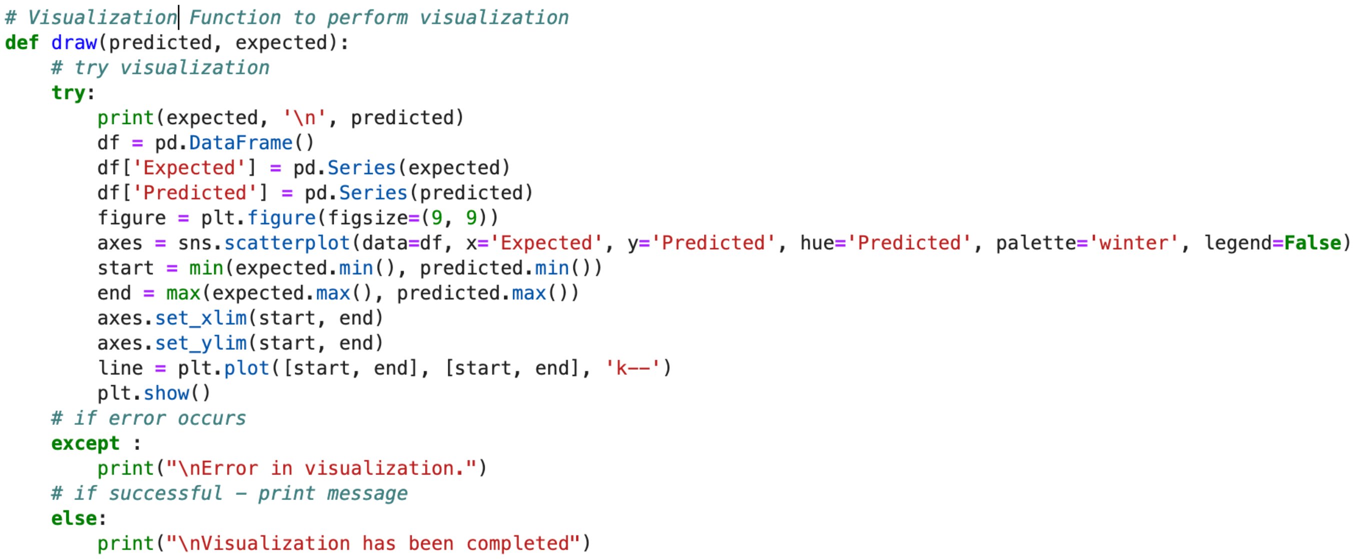
VIII. Performance - Regression Metrics



```
# Regression Metrics : Function to perform the regression metrics defining the performance of the model
def regress(predicted, expected):
    # try regression
    try:
        print('\nCoefficient of Determination : ', metrics.r2_score(expected, predicted))
        print('\nMean Squared Error : ', metrics.mean_squared_error(expected, predicted))
    # if error
    except :
        print("\nError in regression.")
    # if successful - print message
    else:
        print("\nRegression modelling has been done")
```

Snippet 7 Performance using Regression Metrics

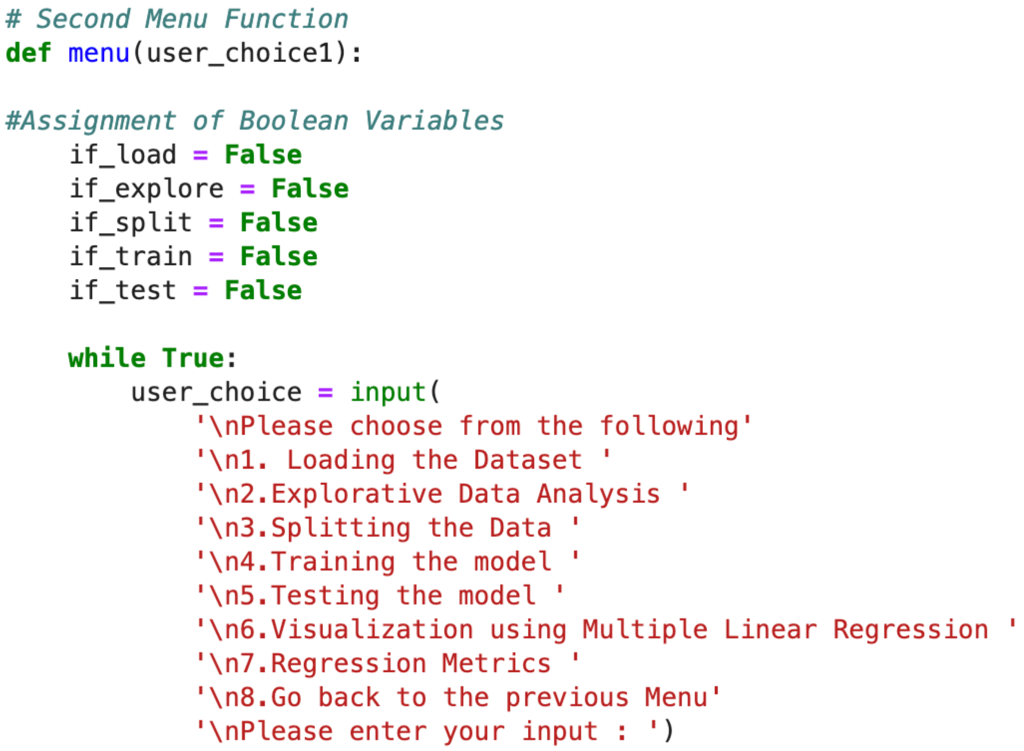
IX. Visualization



```
# Visualization Function to perform visualization
def draw(predicted, expected):
    # try visualization
    try:
        print(expected, '\n', predicted)
        df = pd.DataFrame()
        df['Expected'] = pd.Series(expected)
        df['Predicted'] = pd.Series(predicted)
        figure = plt.figure(figsize=(9, 9))
        axes = sns.scatterplot(data=df, x='Expected', y='Predicted', hue='Predicted', palette='winter', legend=False)
        start = min(expected.min(), predicted.min())
        end = max(expected.max(), predicted.max())
        axes.set_xlim(start, end)
        axes.set_ylim(start, end)
        line = plt.plot([start, end], [start, end], 'k--')
        plt.show()
    # if error occurs
    except :
        print("\nError in visualization.")
    # if successful - print message
    else:
        print("\nVisualization has been completed")
```

Snippet 8 Visualization Function

X. Second Menu Function – To choose from the list & Assignment of the Boolean Variables



```
# Second Menu Function
def menu(user_choice1):

#Assignment of Boolean Variables
    if_load = False
    if_explore = False
    if_split = False
    if_train = False
    if_test = False

    while True:
        user_choice = input(
            '\nPlease choose from the following'
            '\n1. Loading the Dataset '
            '\n2.Explorative Data Analysis '
            '\n3.Splitting the Data '
            '\n4.Training the model '
            '\n5.Testing the model '
            '\n6.Visualization using Multiple Linear Regression '
            '\n7.Regression Metrics '
            '\n8.Go back to the previous Menu'
            '\nPlease enter your input : ')
```

Snippet 9 Second Menu & Boolean Variables

4 REFERENCES

1. Intro to Python for Computer Science and Data Science: Learning to Program with AI, Big Data and The Cloud : 15.5 Case Study: Multiple Linear Regression with the California Housing Dataset

2. SKLEARN datasets library for Diabetes Dataset (TOY DATASET) : https://scikit-learn.org/stable/datasets/toy_dataset.html.

3. SKLEARN Datasets library for California Housing Dataset (REAL WORLD DATASET) : https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_california_housing.html#sklearn.datasets.fetch_california_housing