Homework 6

your name and id

In this homework, you're asked to write the class of **Multivariate** ordinary least square (OLS) regression with Numpy and test its performance with real-world dataset. Please fill the code block cells with your code and comments, run everything (select cell in the menu, and click Run all), save the notebook, and upload it to canvas.

In [1]:
```
# import the packages
import numpy as np
```

Task 1: Define the class for multivariate linear regression

Define a class named `MyLinearRegression` for the multivariate linear regression machine learning problem. It should contain a method called `fit` to estimate parameters, `predict` to generate predictions, and `score` to evaluate performance with R^2 value.

In Task 1, you should write your code with pure Python or Numpy, and are not allowed to use any other packages/functions in Scikit-Learn

Hints:

- For basic structures of this class, you can refer to the single-variable linear regression class defined in lecture notes because they are very similar. You only need to replace the formulas with the mulivariate regression case.
- Please review the mathematical part of lecture notes 10 carefully before writing the code. All the formulas used here are already given in the lecture notes, and you need to pick up the correct formulas to estimate parameters/generate predictions/evaluate performance.
- The most tricky part is about dealing with the intercepts β_0, while we have already done it for you in the `fit` method below.
- For linear algebra operations in Numpy, you can consult [here (https://numpy.org/doc/stable/reference/routines.linalg.html)](https://numpy.org/doc/stable/reference/routines.linalg.html). You can also review TA's discussion notes on Numpy.

```
In [ ]: class MyLinearRegression:
            '''
            your document strings here
            '''

            def fit(self, X, y):
                '''
                your document strings here
                '''

                ones = np.ones((X.shape[0],1)) # column of ones
                X_aug = np.concatenate((ones, X), axis = 1) # the augmented matrix, \tilde{X}
        in our lecture

                # continue your codes

            def predict(self,X):
                '''
                your document strings here
                '''

                # your code here

            def score(self, X, y):
                '''
                your document strings here
                '''

                # your code here
```

Task 2: Application to diabetes dataset

- After defining the class, we can test them with the [diabetes dataset (https://web.stanford.edu/~hastie/Papers/LARS/LeastAngle_2002.pdf)](https://web.stanford.edu/~hastie/Papers/LARS/LeastAngle_2002.pdf), loaded from scikit-learn. We're going to use 10 variables (information about each patient, already mean centered and scaled by the standard deviation) to predict the disease progression (use a number to measure, can be thought as continuous) after one year.

In [2]:
```python
# load the dataset
from sklearn import datasets
db = datasets.load_diabetes()
print(db['DESCR'])
```

```
.. _diabetes_dataset:

Diabetes dataset
----------------

Ten baseline variables, age, sex, body mass index, average blood
pressure, and six blood serum measurements were obtained for each of n =
442 diabetes patients, as well as the response of interest, a
quantitative measure of disease progression one year after baseline.

**Data Set Characteristics:**

  :Number of Instances: 442

  :Number of Attributes: First 10 columns are numeric predictive values

  :Target: Column 11 is a quantitative measure of disease progression one year after
baseline

  :Attribute Information:
      - age     age in years
      - sex
      - bmi     body mass index
      - bp      average blood pressure
      - s1      tc, T-Cells (a type of white blood cells)
      - s2      ldl, low-density lipoproteins
      - s3      hdl, high-density lipoproteins
      - s4      tch, thyroid stimulating hormone
      - s5      ltg, lamotrigine
      - s6      glu, blood sugar level

Note: Each of these 10 feature variables have been mean centered and scaled by the s
tandard deviation times `n_samples` (i.e. the sum of squares of each column totals
1).

Source URL:
https://www4.stat.ncsu.edu/~boos/var.select/diabetes.html

For more information see:
Bradley Efron, Trevor Hastie, Iain Johnstone and Robert Tibshirani (2004) "Least Ang
le Regression," Annals of Statistics (with discussion), 407-499.
(https://web.stanford.edu/~hastie/Papers/LARS/LeastAngle_2002.pdf)
```

- Next, we manully split the traning and test datasets. For the basic concepts, please refer to the lecture notes/discussion files.

In [ ]:
```python
X = db['data'] # already in Numpy array format
y = db['target']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_stat
e=42)
```

- Please use the class you defined to train the linear regression model on the **training dataset**, and report the R^2 on **test dataset**.

In [ ]:
```python
reg = MyLinearRegression()

# continue to write your code here
```

Task 3: Comparison with Scikit-Learn

Repeat the linear regression task above (i.e. train the linear regression on the **training dataset**, and report the R^{2} on **test dataset**.) with calling the methods in [sklearn package (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html)](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html). Ideally, the results should be the same with task 2.

In [ ]:
```
# write your code here
```

Optional Task

1. Create a pandas dataframe of the data and use seaborn to visualize.

1. Can you try the regression module in PyCaret to do the automatic machine learning for this data? You can follow the tutorial [here (https://github.com/pycaret/pycaret/blob/master/tutorials/Regression%20Tutorial%20Level%20Beginner%20-%20REG101.ipynb)](https://github.com/pycaret/pycaret/blob/master/tutorials/Regression%20Tutorial%20Level%20Beginner%20-%20REG101.ipynb)

In [3]:
```
import pandas as pd
data = pd.DataFrame(np.c_[db['data'], db['target']],columns= db['feature_names']+['ta
rget'])
data
```

Out[3]:

	age	sex	bmi	bp	s1	s2	s3	s4	s5	s6	ta
0	0.038076	0.050680	0.061696	0.021872	-0.044223	-0.034821	-0.043401	-0.002592	0.019908	-0.017646	1
1	-0.001882	-0.044642	-0.051474	-0.026328	-0.008449	-0.019163	0.074412	-0.039493	-0.068330	-0.092204	
2	0.085299	0.050680	0.044451	-0.005671	-0.045599	-0.034194	-0.032356	-0.002592	0.002864	-0.025930	1
3	-0.089063	-0.044642	-0.011595	-0.036656	0.012191	0.024991	-0.036038	0.034309	0.022692	-0.009362	2
4	0.005383	-0.044642	-0.036385	0.021872	0.003935	0.015596	0.008142	-0.002592	-0.031991	-0.046641	1
...	...	...	...	...	...	...	...	...	...	...	
437	0.041708	0.050680	0.019662	0.059744	-0.005697	-0.002566	-0.028674	-0.002592	0.031193	0.007207	1
438	-0.005515	0.050680	-0.015906	-0.067642	0.049341	0.079165	-0.028674	0.034309	-0.018118	0.044485	1
439	0.041708	0.050680	-0.015906	0.017282	-0.037344	-0.013840	-0.024993	-0.011080	-0.046879	0.015491	1
440	-0.045472	-0.044642	0.039062	0.001215	0.016318	0.015283	-0.028674	0.026560	0.044528	-0.025930	2
441	-0.045472	-0.044642	-0.073030	-0.081414	0.083740	0.027809	0.173816	-0.039493	-0.004220	0.003064	

442 rows × 11 columns

In [ ]:
```
# write your codes here
```