**MINISTRY OF EDUCATION AND TRAINING**

**THE SAIGON INTERNATIONAL UNIVERSITY**

**FINAL EXAM PROJECT**

**MACHINE LEARNING COURSE**

**TOPIC: BOSTON HOUSING PRICE**

**MAJOR: SOFTWARE ENGINEERING**

**LECTURER:**

DR. HUYNH NGOC TIN

**STUDENTS:**

81012202519 – NGUYEN PHUC TOAN

81012202505 – HONG DAI NGHIA

**HCM City - 1, 2024**

# 

# Table of Contents

[**Introduction 4**](#_heading=h.t23wxjhqchf4)

[**Linear Regression** 5](#_heading=h.w2kqc968lxls)

[1. Introduction 5](#_heading=h.w2kqc968lxls)

[2. Loss Function 5](#_heading=h.w2kqc968lxls)

[3. Cost Function 6](#_heading=h.w2kqc968lxls)

[4. Gradient descent 7](#_heading=h.w2kqc968lxls)

[5. Prediction Function 8](#_heading=h.w2kqc968lxls)

[**Multivariable Linear Regression**](#_heading=h.w2kqc968lxls)

[1. Introduction 8](#_heading=h.w2kqc968lxls)

[2. Loss Function 9](#_heading=h.w2kqc968lxls)

[3. Gradient Descent 10](#_heading=h.w2kqc968lxls)

4[. Prediction Function](#_heading=h.w2kqc968lxls)

5[. Regularization Functions (Hàm Chính quy): 11](#_heading=h.w2kqc968lxls)

[**Logistic regression 12**](#_heading=h.22s1uqjtwsh6)

[1.What is Logistic regression 12](#_heading=h.2et92p0)

[**Overfitting & underfitting 14**](#_heading=h.3dy6vkm)

[1. What is Overfitting & Underfitting ? 14](#_heading=h.yiv371nhgyzl)

[**Regularization 15**](#_heading=h.1t3h5sf)

[1. What is regularization 15](#_heading=h.e7zx2evb7ejf)

[**BOSTON HOUSING DATASET 16**](#_heading=h.g9b2myu58r7i)

[1. Overview 16](#_heading=h.ykb6ahyxlbz5)

[2. About the Dataset 17](#_heading=h.j54dqvy206f8)

[3. Problems 18](#_heading=h.ibchrswgzrvs)

[**MACHINE LEARNING WITH DATASET 19**](#_heading=h.ivoizhsyqq37)

[1. Predict MEDV with Linear Regression 19](#_heading=h.9vrb9uhbo1s0)

[1.1Introduction to Linear Regression for the MEDV Column: 19](#_heading=h.6bto8p4fqc98)

[1.2 CODING 20](#_heading=h.ddjacnxunlle)

[1.2.2 Apply Multivariable Linear Regression and L2 Regularization to Model 21](#_heading=h.oyrjtclrrnya)

[**Logistic Regression with CHAS Column 27**](#_heading=h.5ia96pc82f5m)

[1. Classify CHAS with Logistic Regression 27](#_heading=h.u7xpt5ca7u29)

[1.1Introduction to Linear Regression for the Chas Column 27](#_heading=h.xdeiq9xdmegg)

[1.2 Coding 27](#_heading=h.h61afwies5yn)

[2. Oversampling & Undersampling 31](#_heading=h.frijws1t6par)

[3.Compare results 32](#_heading=h.cnqdaeelg5d)

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# Introduction

Nowadays, AI in general and Machine Learning in particular, both of them are not only a fascinating topic but also a crucial field that profoundly influences various aspects of life and industry. In this report, we will introduce a data set of Bostonhousing.

Within the Boston Housing dataset, providing a predictive model for housing prices using linear regression. Additionally, logistic regression will be employed to address classification challenges inherent in the dataset.

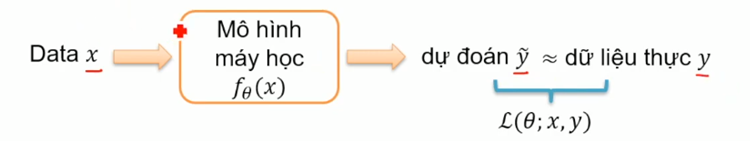
Throughout this brief report, we introduce the Boston Housing dataset, emphasizing the influential features on housing prices. We explore the applications of linear and logistic regression, aiming to swiftly convey key methodologies and concepts. This concise exploration contributes to the broader understanding of AI and ML applications in real-world problem-solving.

## 

# **Linear Regression**

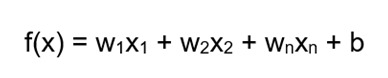
## Introduction

We will have input data, denoted as x, passed through a machine learning model, denoted as f. In this context, f represents the parameters of the model, producing a predicted value y^ that is expected to closely match the actual dataset values denoted as y.



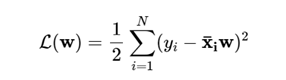
In its simplest form, we can observe that:

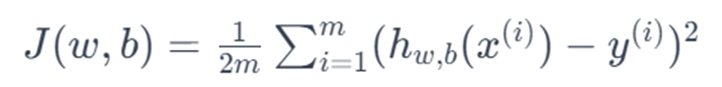
* The larger the house size, the higher the house price.
* The greater the number and size of bedrooms, the higher the house price.
* The farther from the city center, the lower the house price.

The simplest function that can describe the relationship between house price and the three input variables is:

Here, w1, w2,...,wn are constants, and b is also known as bias. Generally, y and y^ represent two different values due to model errors; however, we expect this difference to be very small.

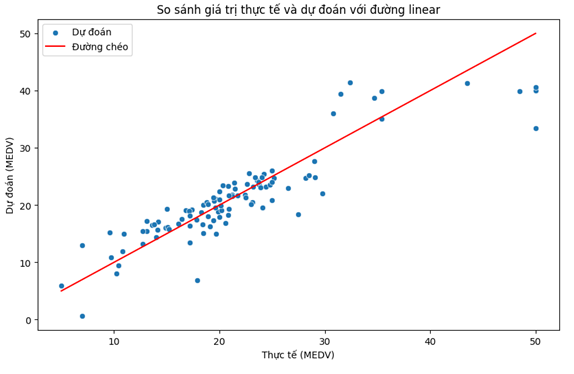
## Loss Function

To assess the similarity between y^ and y, we need a Loss function, which takes parameters denoted as θ (the parameters of the model that will change). The goal is to find values for θ that minimize the Loss function. Taking the example of the Boston housing problem, where the house area is associated with the house price, we aim to find a straight line (depicted in red) that can pass through all data points.However, it's impossible to find a line that passes through all points perfectly. Therefore, we seek the line that minimizes the error across these points. This is achieved by defining a Loss function that quantifies the difference between the predicted values (yi) and the actual values (yi) for each data point. The objective is to find the values of θ that minimize this overall loss. The process involves adjusting the parameters iteratively until the optimal values are found, resulting in a line that best fits the data with the least error:

The commonly used loss function in Linear Regression is Mean Squared Error (MSE), defined as follows:

In which:

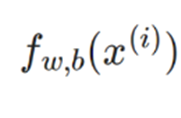
* J(w, b):is the loss function.
* w: is the weight vector.
* b: is the bias term.
* m: is the number of data samples.
* hw,b(x(i)): is the model’s prediction for the i-th sample.
* y(i): is the actual value for the i-th sample.

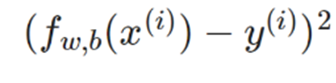


## Cost Function

Our objective is to optimize the model such that the predicted values y^ are as close as possible to the actual values y, measuring how well our model predicts the target price of a house compared to its actual value. To quantify this difference, we use the cost function:

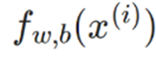
Breaking down each component:

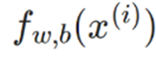
●  : It represents the result of the model's i-th prediction based on the parameters w and b.

● :This is the square of the difference between the model's predicted outcome and the target y(i)).

● The sum of all squared differences is then divided by 2m, where m is the number of training data samples, to calculate the cost value J(w,b).

During training, the following steps occur:

(i) : A prediction is made, also known as a forward pass.

(ii) The difference between the predicted outcome and the true outcome y(i) is calculated and squared.

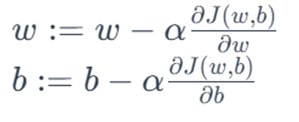
(iii) This squared difference value is added to the overall cost.

Ultimately, the cost function between the parameters w and b is determined by the total cost divided by 2× (the number of training data samples).

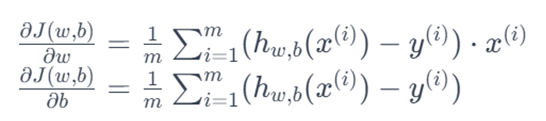
## Gradient descent

The Gradient Descent algorithm is an optimization technique used to update the parameters of a prediction model in order to minimize the loss function. In the context of Linear Regression, our goal is to find the weights and bias that make the model's predictions as close as possible to the actual data.

The Gradient Descent algorithm is employed to update w and b based on the derivatives of the loss function with respect to these parameters. The update formulas can be described as follows:



* α is the learning rate, a constant that determines the learning speed of the algorithm:

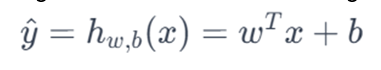


These update formulas guide the adjustment of w and b in the direction that minimizes the value of the loss function, thereby improving the predictive capability of the model.

## Prediction Function

The Prediction Function in Linear Regression is utilized to compute the predicted value y^ for a new data sample x based on the weights w and bias b that have been learned during the model training process. The prediction function is a simple linear equation.

The formula for the prediction function in Linear Regression is:



In which:

* y^: is the predicted value.
* hw,b(x): is the prediction function.
* w: is the weight vector.
* x: is the feature vector of the data sample.
* b: is the bias term.

Once the model has been trained, we can use the prediction function to make new predictions for data samples that were not seen during the training process. This allows us to estimate the predicted values y^ for new data points based on the linear relationship learned from the training data.Specifically, this process typically involves substituting the values of x into the equation and using the learned weights and bias to compute the predicted value y^. The result is a linear prediction for the target value based on the features of that data sample.

## 

# Multivariable Linear Regression

## Prediction Function

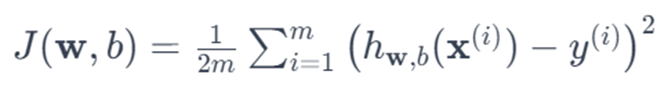
In Multivariable Linear Regression, we extend the principles of Linear Regression to accommodate multiple input variables, providing a more comprehensive model for predicting an outcome. In this context, we have input data denoted as x, which consists of multiple features, passed through a machine learning model denoted as f. Similar to the single-variable case, f represents the parameters of the model, producing a predicted value y^ that aims to closely align with the actual dataset values denoted as y.

Expanding beyond the simplicity of single-variable relationships, Multivariable Linear Regression enables us to consider a broader set of factors influencing the target variable. For instance, when predicting house prices, the model may now take into account not only the size of the house but also the number of bedrooms, the distance from the city center, and potentially other relevant features. The goal remains to identify the optimal weights (w1, w2,...,wn ) and bias (b) that minimize the difference between the predicted values (y^) and the actual values (y).

The Multivariable Linear Regression function can be expressed as:

Here, w1, w2,...,wn are the respective weights for each feature, x1, x2,...,xn, and b is the bias term. The relationship between y and y^ acknowledges the presence of multiple input variables, and despite inherent model errors, the objective remains to minimize the discrepancies between the predicted and actual values.

## 2. Loss Function

The Multivariable Loss Function in Multivariable Linear Regression is often defined using Mean Squared Error (MSE). MSE calculates the average of the squared differences between the predicted values and the actual values across the entire training dataset. The formula for MSE in Multivariable Linear Regression is:

in which:

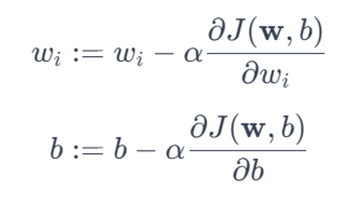
* J(w,b) is a loss function.
* m is the number of data samples.
* hw,b(x(i)) is the model’s prediction for the i-th sample.
* y(i) : is the actual value for the i-th sample.

The objective during the training process is to minimize the loss function J(w,b) by adjusting the weights w and b of the model. This process is typically carried out using optimization methods such as Gradient Descent.

It is important to note that dividing by 2m in the MSE formula is for simplifying the calculation of derivatives and does not impact the optimization process significantly. When applying the Gradient Descent algorithm, the derivative is divided by 2m for computational convenience, ensuring a smoother optimization process.

## 3. Gradient Descent

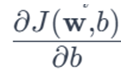
Multivariable Gradient Descent is used to update the weights and bias of the Multivariable Linear Regression model to minimize the loss function. The derivatives of the loss function with respect to each weight wi and the bias term b are utilized to adjust their values. The update formula for each step of Gradient Descent is:



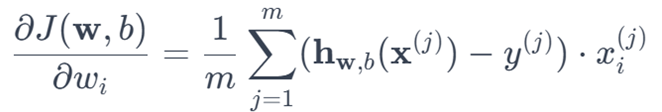
Here:

α is the learning rate, a constant determining the step size in each iteration.

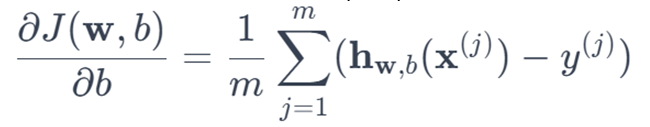
 : is the partial derivative of the loss function J(w,b) with respect to wi

: is the partial derivative of the loss function J(w,b) with respect to b.

The derivative of the loss function J(w,b) with respect to wi is calculated as follows:



The derivative of the loss function J(w,b) with respect to b is calculated as follows:



This process is repeated until the loss function reaches an optimal value or after a certain number of update steps.

## 4. Prediction Function

The Multivariable Prediction Function in Multivariable Linear Regression is utilized to predict the output value (y) based on multiple features (x1, x2,...,xn). This function is described by a linear equation.

The Multivariable Prediction Function takes the following form:

* hw,b(x) is the predicted value.
* w is the weight vector, where each element wi is associated with the feature xi
* x is the feature vector of the data sample, where each element xi represents the corresponding feature value.
* b is the bias term.

In this equation, we perform a matrix multiplication between the weight vector w and the feature vector x, then add the bias term b. The result is a numerical value, an estimate by the model for the output value (y).

The objective during the training process is to find the values of the weight vector w and bias term b such that the loss function (typically Mean Squared Error) is minimized. This means the model produces predictions that are as close as possible to the actual values in the training dataset. This is often achieved using optimization methods such as Gradient Descent.

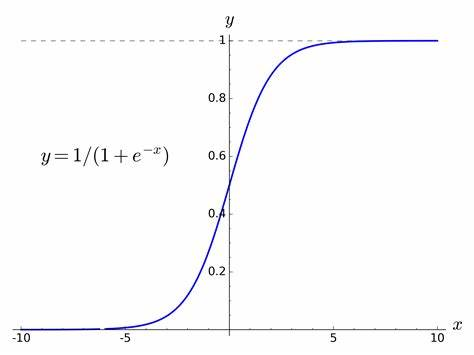
# Logistic regression

.

## 1.What is Logistic regression

Nowadays, AI in general and Machine Learning in particular, both of them are not only a fascinating topic but also a crucial field that profoundly influences various aspects of life and industry. In this report, we will introduce the flower topic, specifically here is about Iris flowers.

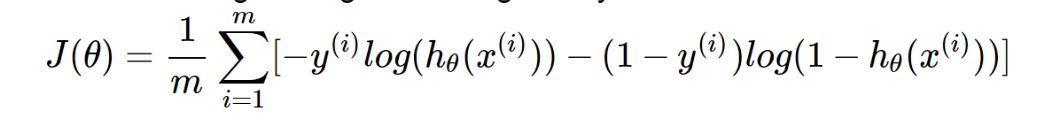
As you know, what we had learned before about Linear Regression. That model is used to predict a continuous outcome like price, age, size…Unlike LR, Logistic Regression used to predict a binary classification something like 0 or 1, true or false, has disease or not disease. So the question is how can it work ? Let me introduce you this beautiful function, the sigmoid function



It looks easy right? You just put the x in the sigmoid function if it is more than 0.5 then its true if less than 0.5 its false. go the next point, the x determined by this

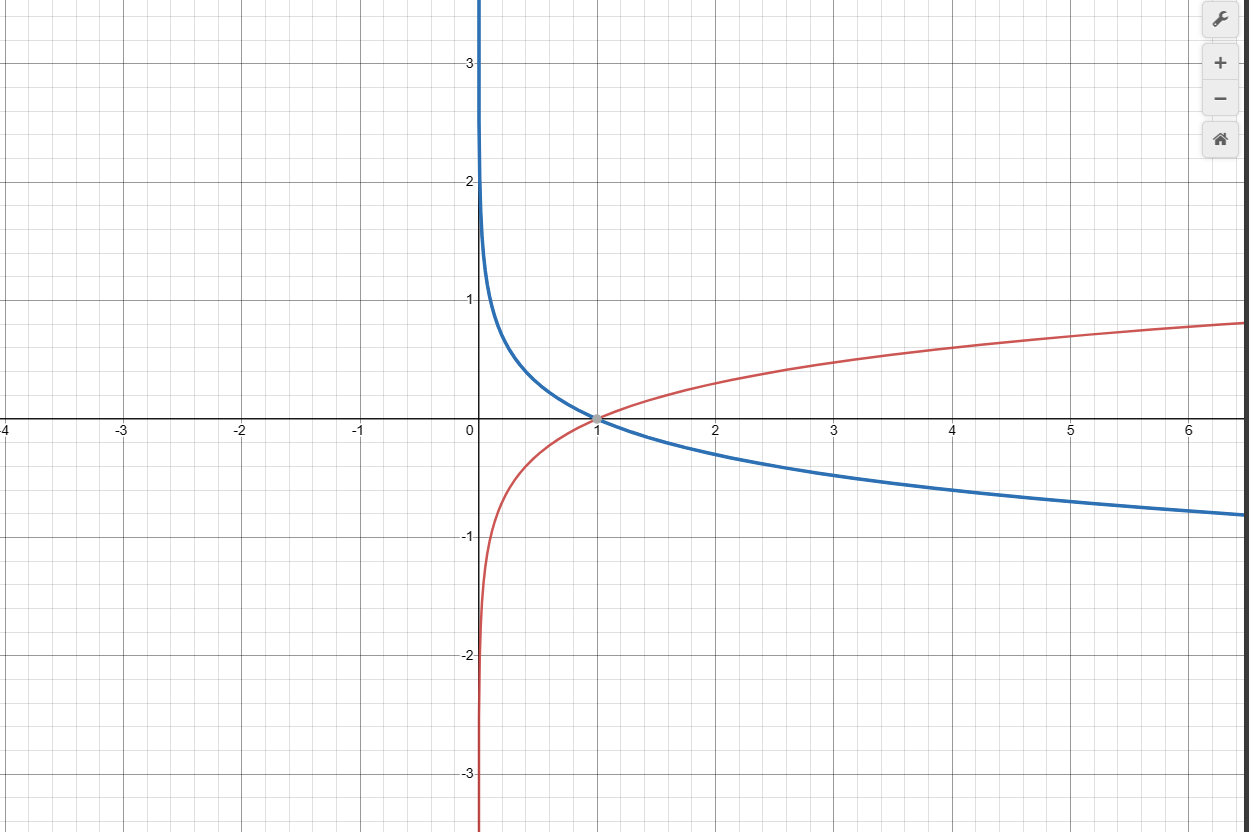
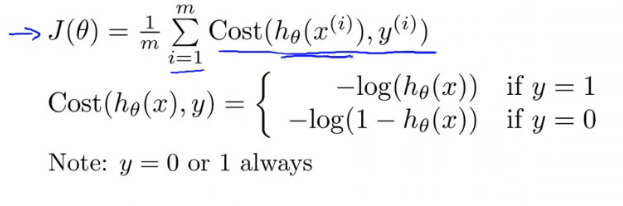


Like Linear Regression we have cost function too

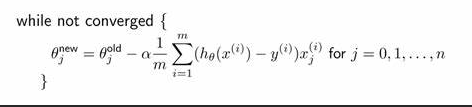


Our goal to optimize the cost function to find the best parameter values means weights for the logistic regression model. This is the way how in work its Gradient Descent

In logistic regression y only has two values: 1 or 0. So in the case y = 0,if the predict close to 0 that means log(1-y(predict)) close to 1 and its lead to have less value. In the case y = 1, if the predict is close to 1 then the error will be less. Also if the predict close to 0 mean far from the actual value the error will be high.



With the gradient decent repeat and update the W.

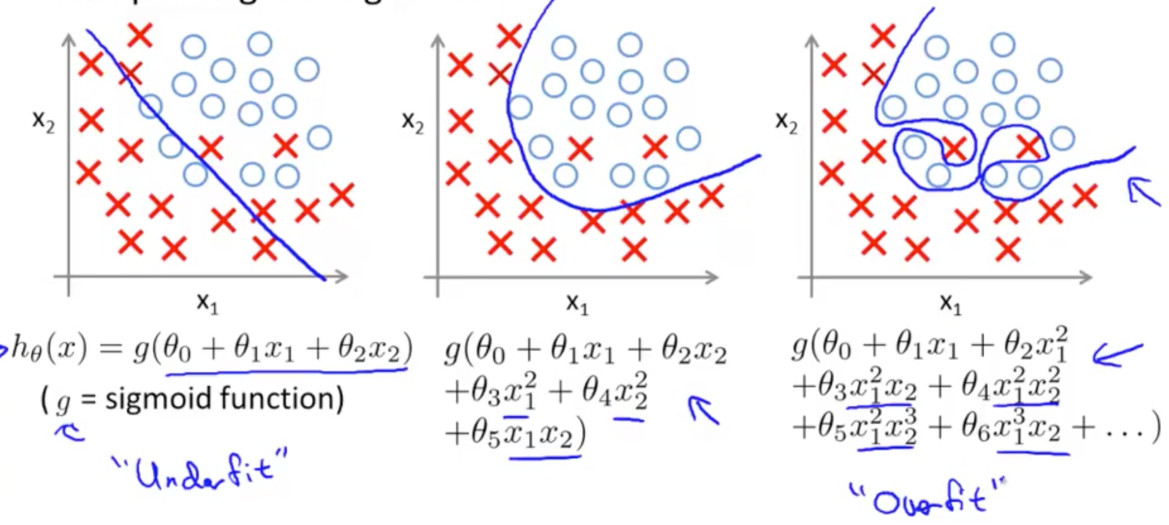


So that is about Logistic Regression and it's just for binary classification.but with this dataset, we have 3 things to classify right? This is called multi-class classification problems. for this problem, we use another strategy for logistic regression; it's also known that the OVR stands for One versus Rest . Its extending binary classification algorithms

# Overfitting & underfitting

## What is Overfitting & Underfitting ?

In process training we may meet overfitting & underfitting. So what that is let’s find out.



In the left is the underfitting and in the right is the overfitting of classification in logistic regression. These situations happen when the dataset has noise and. Or even maybe the size of the training data is not enough.

About the overfitting, creating the decision boundaries highly fit with the dataset, it might fit the noise of information. This is the high variance model like you predict training data very well but can’t predict test data well.

The reason for overfit may be because you cannot control the training data. It seems like training data is one type pattern and test data is another pattern that your model doesn’t fit.

Underfitting that is your model is too simple so this makes your model predict both train and test is bad .

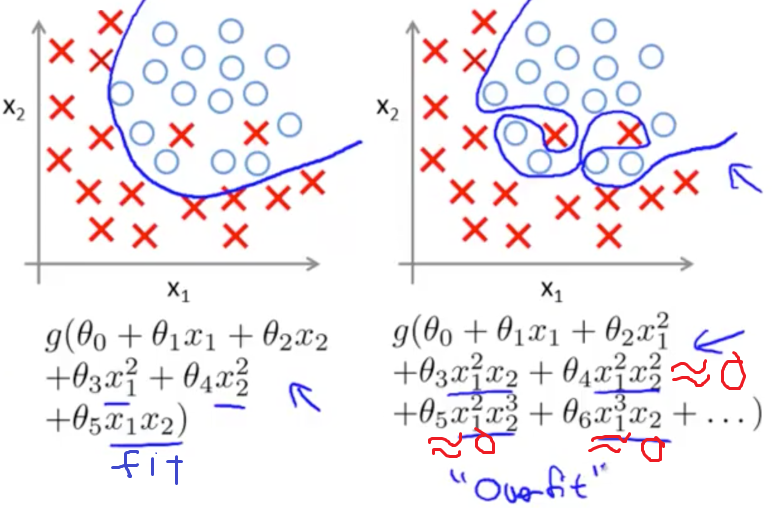
bad.So how to fix the problem. Some solutions collect more data, select features to keep or change another algorithm.

Or to avoid the overfitting we can visualize the training data to analyze that to suit what we want. But we often use regularization.

# Regularization

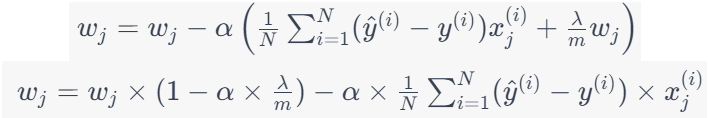
## What is regularization

Now we come to regularization, this is the way how do we use to solve the overfitting problem.



On the left is the good so the way we make the overfit to good fit is by making the theta4(weight4), theta(weight)5,theta(weight)6 is nearly to 0. So in normal we are using gradient descent to get a new weight.

But that way make weight depend on y and x a lot and even if loss not change anymore so this make w will not change anymore. So we need to add a penalty for weight by the new formula.



So in the first formula we can multiply alpha to these numbers inside and then take wj out so we get the new formula just the transform.

So take an example as **Lambda = 1**, alpha = 0.01, m =50

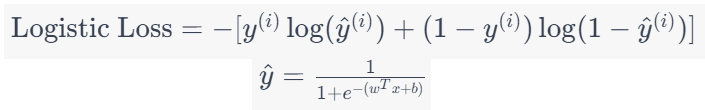
⇒ wj = wj \* (1-0.01\*(1/50)) - Loss = **0.998wj** - Loss

The second example is **Lambda = 5000**, alpha =0.01, m =50

⇒ wj = wj\*(1- 0.01\*(5000/50)) - Loss = **0wj** - Loss

⇒ So the bigger lambda means that the wj will get to 0 faster. And bias is the same with weight.

Assump weight and bias is 0 after lambda 5000 so we put weight and bias to the loss function:



And because loss depends on weight and bias so when weight and bias is nearly 0 and loss will be 0. So make it a good fit as what we want in the beginning.

# BOSTON HOUSING DATASET

# 

## Overview

The Boston Housing Dataset is a seminal collection of data frequently employed in the realms of machine learning and statistical analysis. Originating from a study conducted in 1978 by researchers Harrison and Rubinfeld, the dataset comprises information on 506 suburbs of Boston, each characterized by 13 distinct features. These features encompass a wide array of socioeconomic and environmental factors, ranging from crime rates and zoning proportions to air quality and population demographics.

Key attributes of the dataset include the per capita crime rate, residential land zoning details, nitric oxide concentration, average number of rooms per dwelling, and various other indicators. Integral to the dataset is the inclusion of a target variable: the median value of owner-occupied homes, denominated in thousands of dollars.

Widely recognized for its application in regression analysis, the dataset serves as a foundational resource for machine learning practitioners and researchers. The overarching goal is often to leverage the provided features to predict the median home value, making it an excellent benchmark for regression model development and evaluation. The dataset is easily accessible through popular machine learning libraries like scikit-learn, enhancing its utility in both educational and practical contexts.

Despite its age, the Boston Housing Dataset maintains its relevance and popularity due to its well-documented structure and continued applicability in real-world problem-solving. It has played a pivotal role in studies exploring the intricate relationships between diverse factors and housing prices. The dataset's enduring presence underscores its significance as a valuable tool for understanding the complexities of real estate valuation and the development of predictive algorithms

## About the Dataset

The Boston Housing Dataset is a significant resource in the fields of machine learning and statistics. It contains information about 506 suburbs of the city of Boston and is widely used in studies on regression analysis and predictive modeling. Below is an introduction to the columns and indices of the dataset:

1. CRIM (Per capita crime rate by town):

- Measures the crime rate in each town.

2. ZN (Proportion of residential land zoned for lots over 25,000 sq. ft.)

- Determines the proportion of land planned for residential construction on large lots.

3. INDUS (Proportion of non-retail business acres per town):

- Indicates the proportion of land used for non-retail business activities.

4. CHAS (Charles River dummy variable):

- A dummy variable, equal to 1 if the area borders the Charles River and 0 otherwise.

5. NOX (Nitric oxides concentration):

- Measures the concentration of nitric oxides in the air.

6. RM (Average number of rooms per dwelling):

- Represents the average number of rooms in each dwelling.

7. AGE (Proportion of owner-occupied units built prior to 1940):

- Specifies the proportion of owner-occupied units built before 1940.

8. DIS (Weighted distances to five Boston employment centers):

- Measures the weighted distances to five employment centers in Boston.

9. RAD (Index of accessibility to radial highways):

- An index evaluating the accessibility to radial highways.

10. TAX (Full-value property tax rate per $10,000):

- Determines the full-value property tax rate for each $10,000 value of the home.

11. PTRATIO (Pupil-teacher ratio by town):

- Indicates the pupil-teacher ratio by town.

12. B (1000(Bk - 0.63)^2), where Bk is the proportion of Black residents by town:

- An index related to the proportion of Black residents in the town.

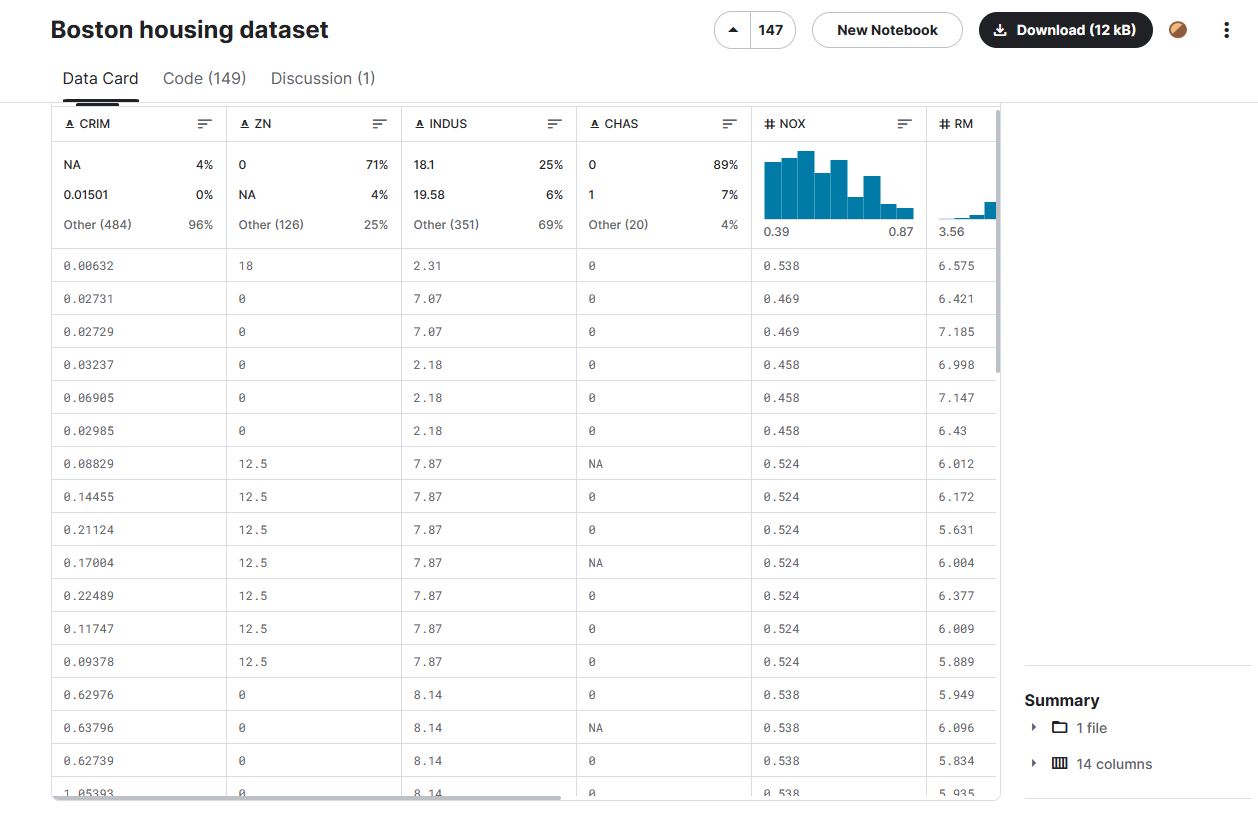
13. LSTAT (Percentage of lower status of the population):

- Specifies the percentage of the population with lower socioeconomic status.

14. MEDV (Median value of owner-occupied homes in $1000s):

- The target variable, representing the median value of owner-occupied homes in thousands of dollars.

The Boston Housing Dataset provides an effective opportunity to explore the relationships between various factors and housing prices. It serves as a platform for the development and evaluation of regression models in the field of machine learning.



Source : [Boston housing dataset (kaggle.com)](https://www.kaggle.com/datasets/altavish/boston-housing-dataset)

## Problems

To address the challenges within the dataset, we will employ two primary methods: linear regression to predict values in the MEDV column (Median value of owner-occupied homes), and logistic regression to classify the CHAS variable (Charles River dummy variable).

1. Linear Regression for the MEDV Column:

- We will utilize linear regression to build a model for predicting the average value of owner-occupied homes (MEDV) based on various features of suburban areas. This model can offer a comprehensive understanding of how different factors influence housing prices.

2. Logistic Regression for the CHAS Variable:

- For the CHAS variable, we will use logistic regression to construct a classification model for proximity to the Charles River. This model will help us gain insights into how geographical factors, such as location, may impact the likelihood of an area bordering the Charles River.

Combining both of these approaches will provide us with a multidimensional and detailed view of the relationships between different factors in the dataset and the value of homes. Simultaneously, using these models can assist in making predictions and classifications on new data, aiding in a deeper understanding of the determinants of housing values and proximity to the Charles River in Boston's suburban areas

.

# MACHINE LEARNING WITH DATASET

## Predict MEDV with Linear Regression

### 1.1Introduction to Linear Regression for the MEDV Column:

Linear regression is a powerful statistical technique employed to model the relationship between a dependent variable and one or more independent variables by fitting a linear equation to the observed data. In the context of the Boston Housing Dataset, we will apply linear regression specifically to predict the median value of owner-occupied homes (MEDV). This modeling approach assumes a linear relationship between the features associated with suburban areas and the corresponding housing prices.

The primary objective of employing linear regression on the MEDV column is to create a predictive model that can estimate the median home value based on various features such as crime rates, zoning proportions, air quality, and socioeconomic indicators. The resulting model will be a mathematical representation of how changes in these features relate to changes in the median home value.

Through the process of training the linear regression model, we aim to identify and quantify the impact of individual features on housing prices. This understanding can be instrumental in making informed decisions related to real estate, urban planning, and socioeconomic development.

By leveraging the linear regression technique, we seek not only to predict median home values but also to gain insights into the underlying patterns and trends within the dataset. The model's coefficients and statistical metrics will provide valuable information about the strength and direction of the relationships, contributing to a comprehensive analysis of the factors influencing housing prices in the Boston suburban areas.

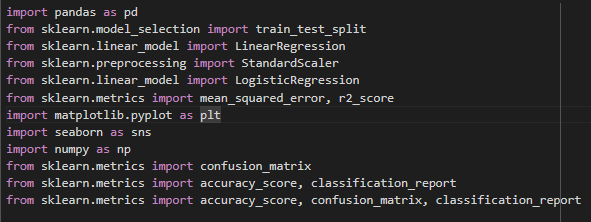
### 1.2 CODING

We will use Python and Colaboratory to code and show the result

Python is a versatile and powerful programming language that lets you work quickly and integrate systems more effectively with many libraries that support in calculation, especially math. With many tools to address mathematics, python is used to solve many machine learning problems.

Colaboratory, or Colab for short, is a free online service that allows you to create and run Python code cells with rich, interactive outputs. Colab is hosted by Google and runs entirely in the cloud, so you don’t need to install anything on your local machine.

1.2.1 Libraries



1.pandas (pd):

Purpose: Pandas is a powerful data manipulation and analysis library. It provides data structures like DataFrame for handling structured data and Series for one-dimensional data.

Usage in Code: Used for reading and manipulating datasets, such as loading data into a DataFrame.

2.scikit-learn:

Purpose: Scikit-learn is a versatile machine learning library in Python. It offers a wide range of tools for tasks such as classification, regression, clustering, and model selection.

Usage in Code: Utilized for model training, evaluation, and preprocessing.

3.matplotlib.pyplot (plt):

Purpose: Matplotlib is a popular plotting library in Python. Pyplot is a module within Matplotlib that provides functions for creating static, animated, and interactive plots.

Usage in Code: Used for visualizing data and model results.

4.seaborn (sns):

Purpose: Seaborn is a statistical data visualization library built on top of Matplotlib. It provides an interface for creating attractive and informative statistical graphics.

Usage in Code: Used for enhancing the visual appeal of plots.

5.numpy (np):

Purpose: NumPy is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with mathematical functions to operate on these data structures.

Usage in Code: Utilized for numerical operations and array manipulations.

6.Confusion Matrix and Classification Report:

Purpose: These are part of scikit-learn and are used for evaluating classification models.

Usage in Code: Employed to assess the performance of the Logistic Regression model, including accuracy, precision, recall, and F1-score

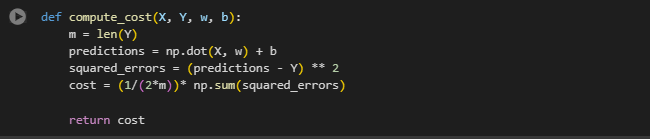
### 

### 1.2.2 Apply Multivariable Linear Regression and L2 Regularization to Model

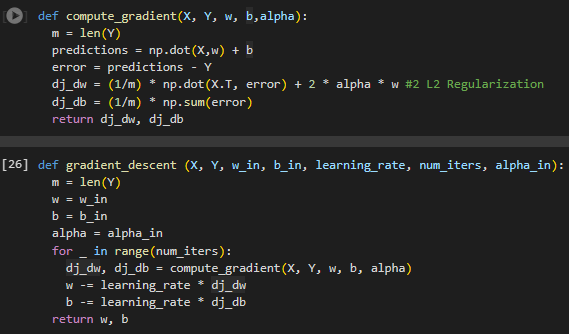
Because our dataset has numerous features, we will apply Multivariable Linear Regression instead of Linear Regression and also use L2 Regression .

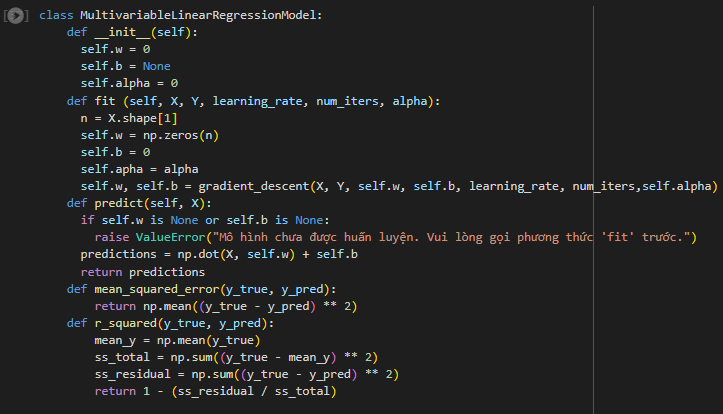
Here is the code Multivariable Linear Regression for this dataset to predict MEDV

First we will built cost function :

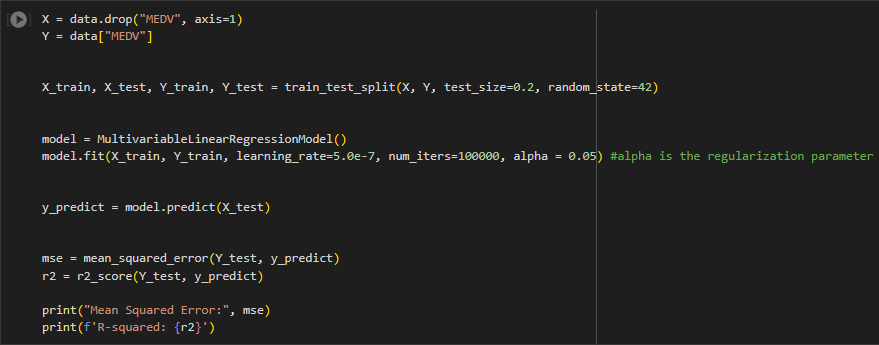


Gradient Descent

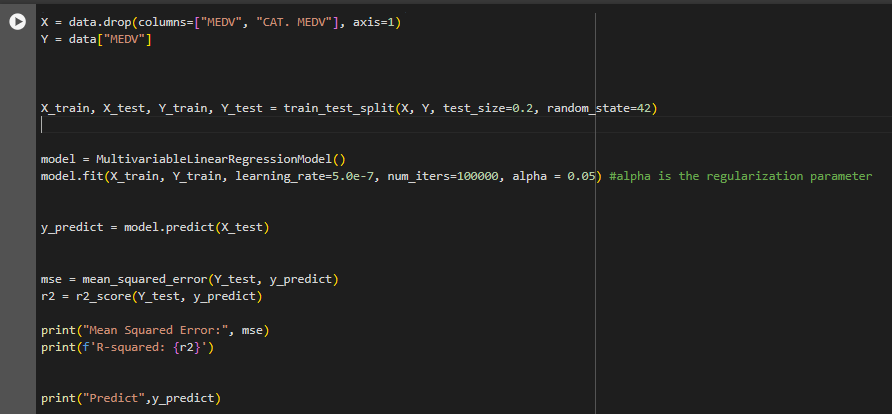


Multivariable Linear Regression   
  


And then will use the dataset to train the model.



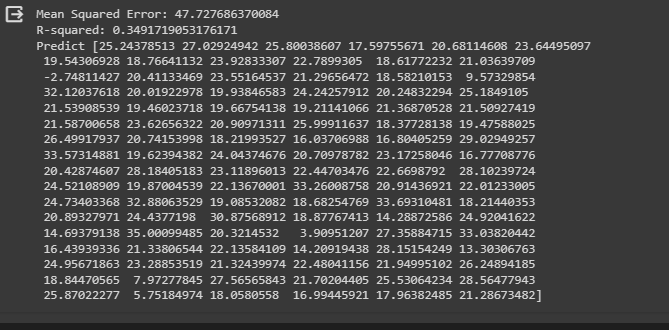
Result



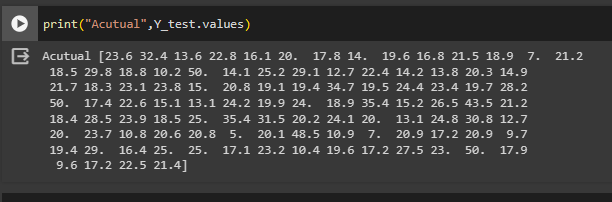
* Mean Squared : MSE, or Mean Squared Error, is a measurement of the quality of a regression model. It quantifies the magnitude of squared errors between predicted and actual values
* R2 : used to evaluate the performance of a regression model. Unlike Mean Squared Error (MSE), R2 provides a measure of how well the independent variable(s) explain the variance in the dependent variable

As you can see, our model the figure of MSE quite high while R-squared figures extremely low which mean the Model we have train is not good at we expected. Here is our predict and actual value.

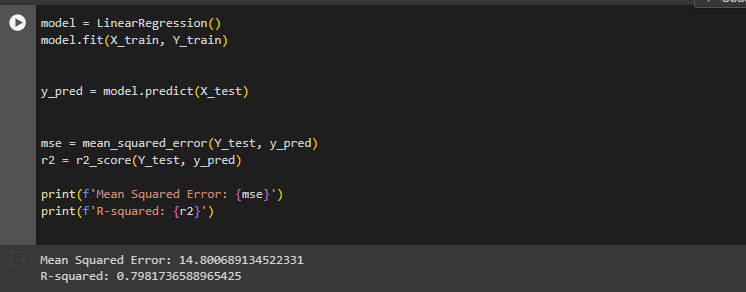
Predictions



Actual Values



And this is the result when we using library from sklearn to solve our problem with Linear Regression

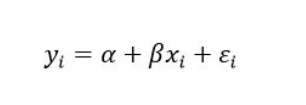


The result is significantly improved so the question is How it can be?

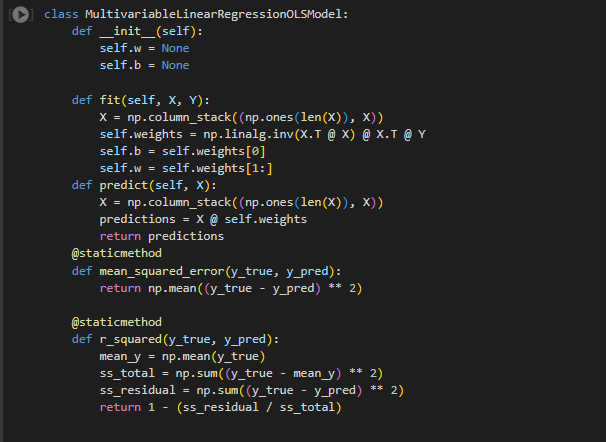
We have study and figure it out is that the library use the equation call OLS (Ordinary Least Squared)

The Ordinary Least Squares (OLS) method is a widely used approach in statistics and econometrics to estimate the parameters of a linear regression model. In a linear regression model, the goal is to find the coefficients that minimize the sum of the squared differences between the observed and predicted values. The OLS method provides a systematic way to determine these coefficients

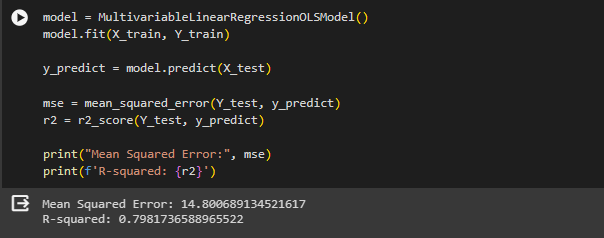
The general form of a linear regression model is given by



So we apply the OLS equation to our model by like this



and this is our result when we get after applying the OLS equation to Model



OLS was highly improved model and reduce the mean Squared Error also the figures of R-squared get better

# Logistic Regression with CHAS Column

## Classify CHAS with Logistic Regression

### 1.1Introduction to Linear Regression for the Chas Column

The "CHAS" column in the Boston Housing dataset is a dummy variable representing the Charles River. This binary variable takes on the value of 1 if the district (row) borders the Charles River and 0 if it does not. In other words:

CHAS (Charles River)

- ValuCharles River and those that are not. In a broader context, it may serve as a proxy for geographic e 1: District borders the Charles River.

- Value 0: District does not border the Charles River.

The CHAS column is used to indicate the distinction between districts that are located along the location characteristics that could impact housing prices. The use of dummy variables is common in regression analysis to represent categorical data, and in this case, it helps capture the influence of proximity to the river on housing characteristics.

In the context of the Boston Housing dataset, Logistic Regression can be employed to classify the "CHAS" column, which represents whether a district is adjacent to the Charles River. The logistic regression model estimates the probability that a given district borders the river based on various features such as crime rate, zoning, air quality, and others.

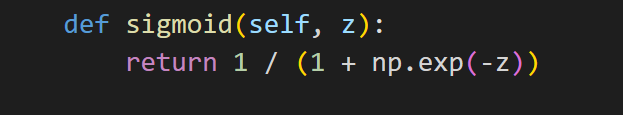
The logistic regression model employs the logistic function (sigmoid) to map the linear combination of input features to a probability between 0 and 1. This probability is then thresholded to make a binary prediction. During training, the model adjusts its parameters to maximize the likelihood of observing the actual outcomes in the training data. After training, the model can be used to predict whether a district is likely to be near the Charles River or not.

Evaluation metrics like accuracy, precision, recall, and the confusion matrix can be used to assess the model's performance in classifying the "CHAS" column. Logistic Regression provides a straightforward and interpretable way to model the relationship between input features and the probability of a binary outcome, making it a valuable tool for binary classification tasks in various domains

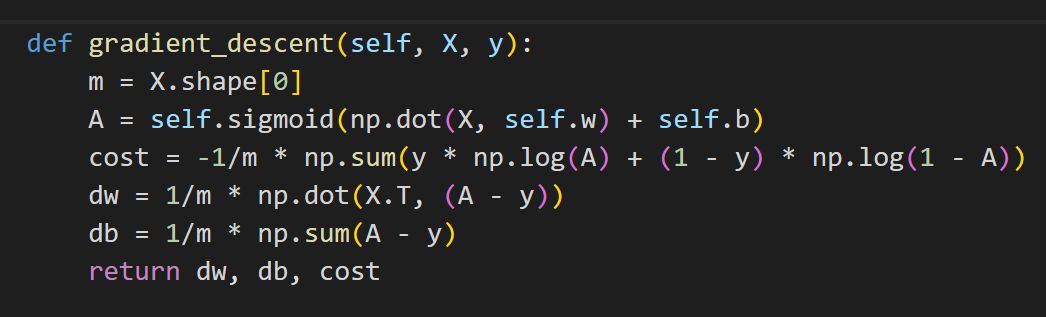
### 1.2 Coding

Logistic regression Model :

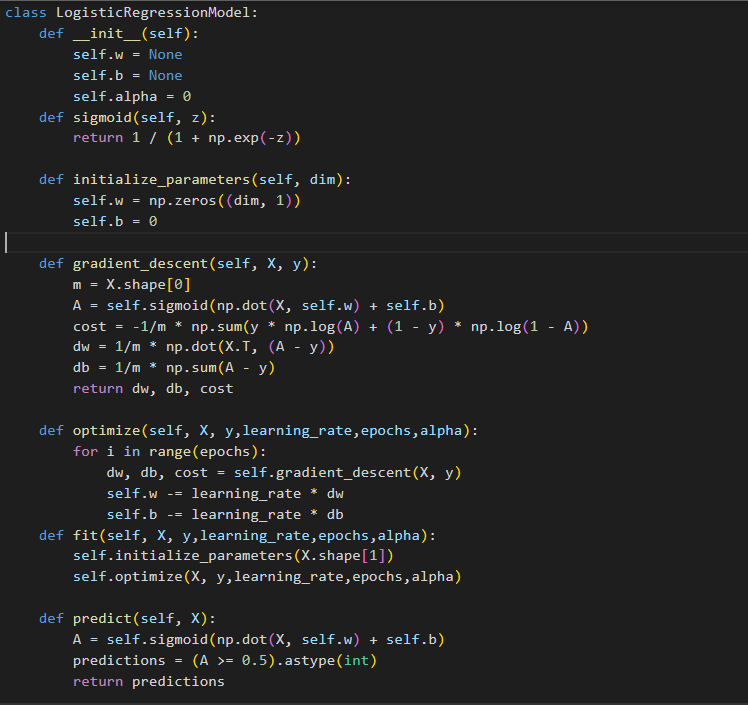
1. Build Sigmoid Function



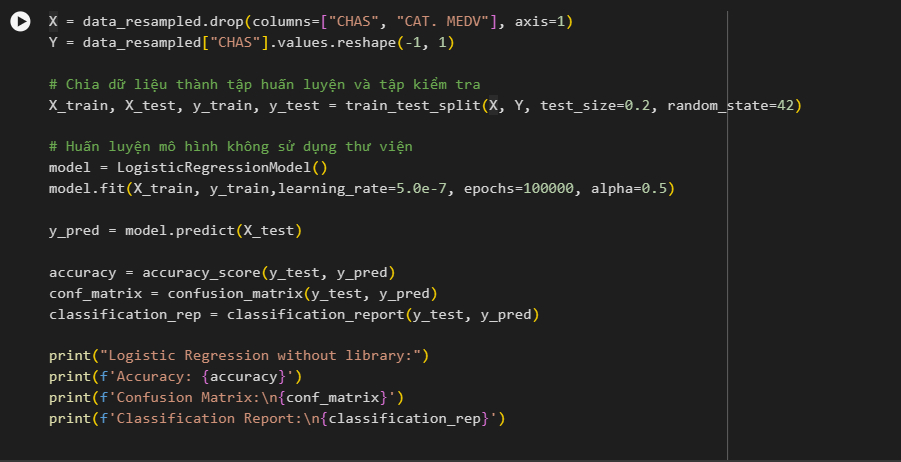
1. Gradient Descent :



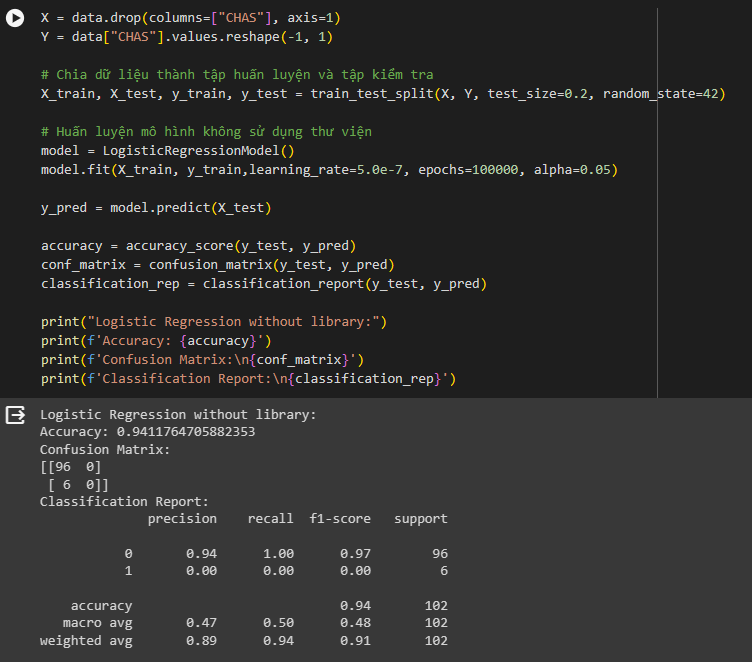
1. Model :



1. Split data to train the model :

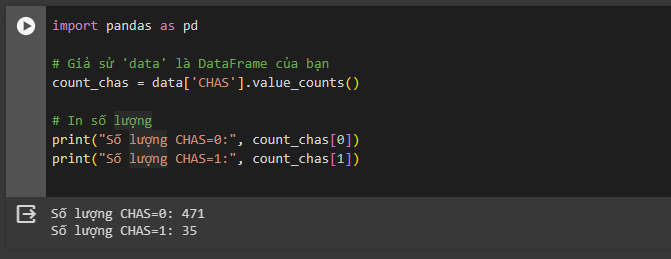


4. Result

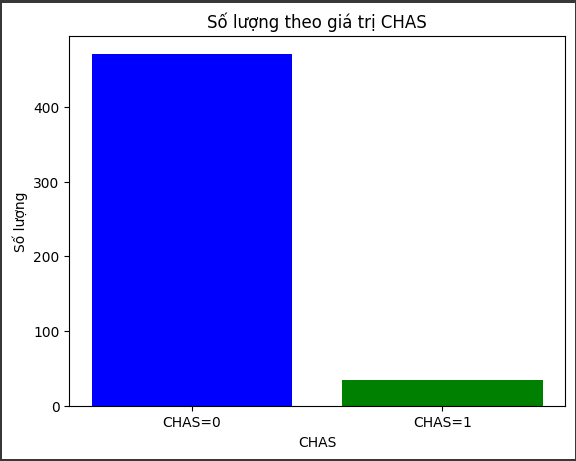


We can see that the other precision is high and when looking at the report, the precision of 0 is 0.94 but for 1 it is only 0.0. What is the reason?

Let's try to look inside the data set of the Chas column.



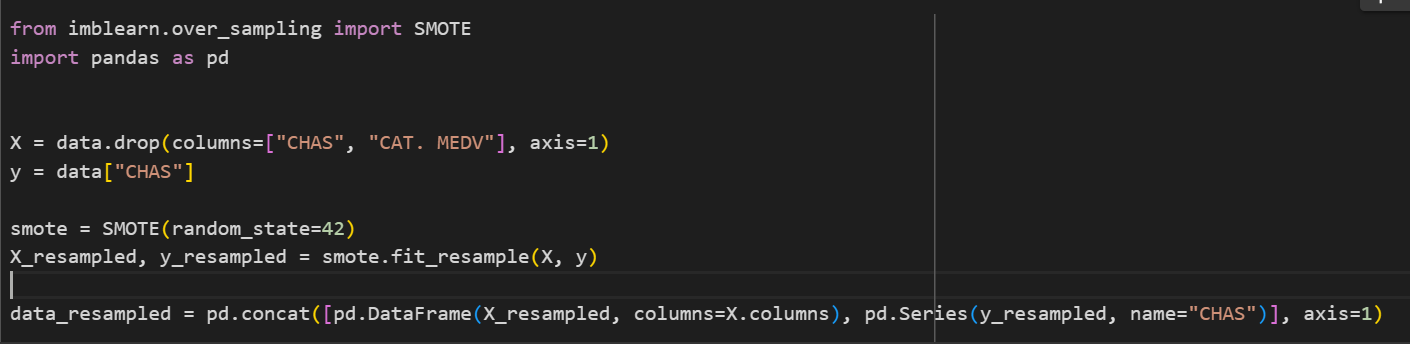
We can see that the huge difference in the number of values in the chas column has caused an imbalance in the data and made it difficult for machine learning to learn from the data. So how to solve this problem?



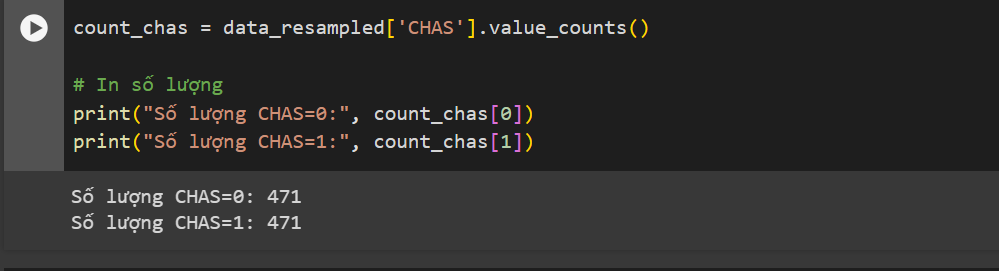
## 2. Oversampling & Undersampling

In machine learning, oversampling involves increasing instances in the minority class to balance the dataset, often using techniques like SMOTE. Conversely, undersampling reduces instances in the majority class to achieve balance. Both approaches address imbalanced datasets, enhancing model performance in classification tasks. Careful implementation and evaluation are essential to avoid pitfalls such as overfitting or loss of information.

Here we will process the data using smote library:

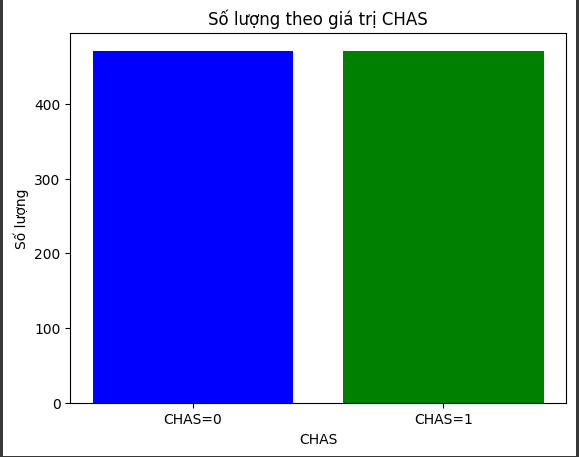


Chas column after being processed :



When using SMOTE (Synthetic Minority Over-sampling Technique), additional columns are processed in a similar manner to the original columns during the synthetic sample generation process. SMOTE not only generates synthetic samples for the target variable but also creates synthetic values for each individual feature (column).

SMOTE creates synthetic samples by randomly selecting one or more neighboring instances from the minority class and generating new synthetic samples based on linear combinations of those instances. This process is applied independently to each feature, ensuring that synthetic values are generated for each feature separately.



## 

## 

## 

## 

## 

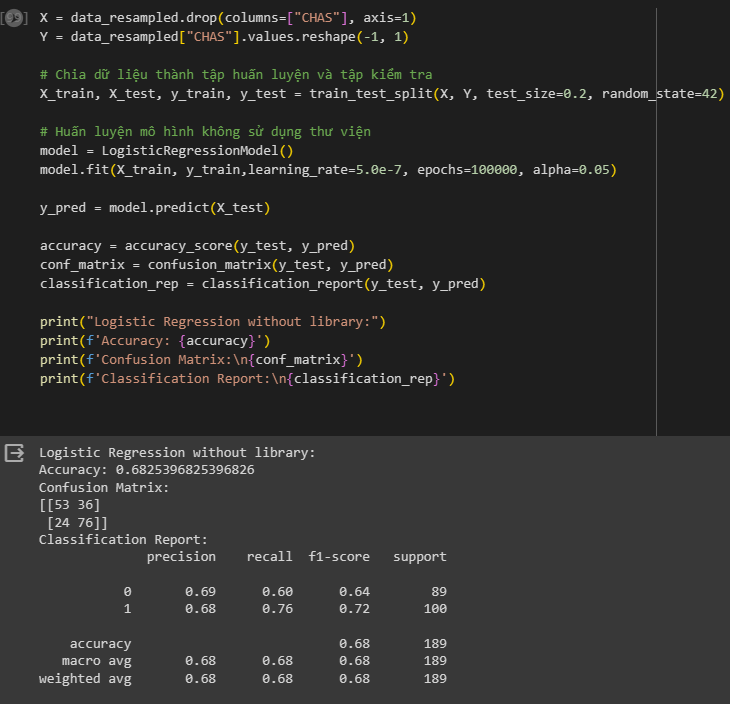
## 

## 

## 

## 3.Compare results

Here is the result of preprocessing the chas column data:



Có thể thấy kết quả đã được cải thiện nhiều sau khi xử lý dữ liệu. Vậy hãy so sánh thêm kết quả khi sử dụng thư viện

