

# Linear Regression

A unit of Machine Learning



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Contents

- ❖ Simple Linear Regression
- ❖ Multiple Linear Regression



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Simple linear regression

## What is regression?

- ❖ The main goal of regression is the construction of an efficient model to predict the dependent attributes from a bunch of attribute variables. A regression problem is when the output variable is either real or a continuous value i.e salary, weight, area, etc.
- ❖ We can also define regression as a statistical means that is used in applications like housing, investing, etc. It is used to predict the relationship between a dependent variable and a bunch of independent variables. Let us take a look at various types of regression techniques.



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Simple linear regression

## Types of regression?

- ❖ Simple Linear Regression
- ❖ Polynomial Regression
- ❖ Support Vector Regression
- ❖ Decision Tree Regression
- ❖ Random Forest Regression



**Prof. Trupesh Patel**

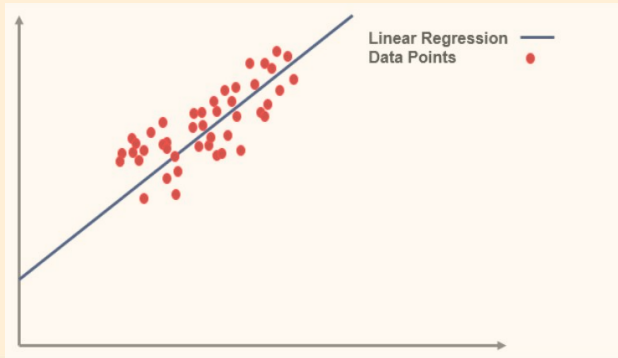
*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# What is Linear regression?

- ❖ Simple linear regression is a regression technique in which the independent variable has a linear relationship with the dependent variable.
- ❖ The straight line in the diagram is the best fit line.
- ❖ The main goal of the simple linear regression is to consider the given data points and plot the best fit line to fit the model in the best way possible.



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*

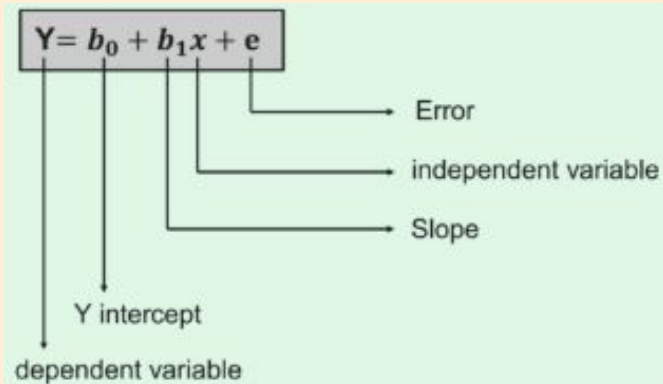


*trupesh1991@gmail.com*

# Linear regression terminology

## ❖ Cost function :

- ❖ The dependent variable that is to be predicted is denoted by  $Y$ .
- ❖ A line that touches the y-axis is denoted by the intercept  $b_0$ .
- ❖  $b_1$  is the slope of the line,  $x$  represents the independent variables that determine the prediction of  $Y$ .
- ❖ The error in the resultant prediction is denoted by  $e$ .



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Linear regression terminology

## ❖ **Cost function :**

- ❖ The cost function provides the best possible values for  $b_0$  and  $b_1$  to make the best fit line for the data points. We do it by converting this problem into a minimization problem to get the best values for  $b_0$  and  $b_1$ . The error is minimized in this problem between the actual value and the predicted value.
- ❖ We choose the function above to minimize the error. We square the error difference and sum the error over all data points, the division between the total number of data points. Then, the produced value provides the averaged square error over all data points.
- ❖ It is also known as MSE(Mean Squared Error), and we change the values of  $b_0$  and  $b_1$  so that the MSE value is settled at the minimum.



**Prof. Trupesh Patel**

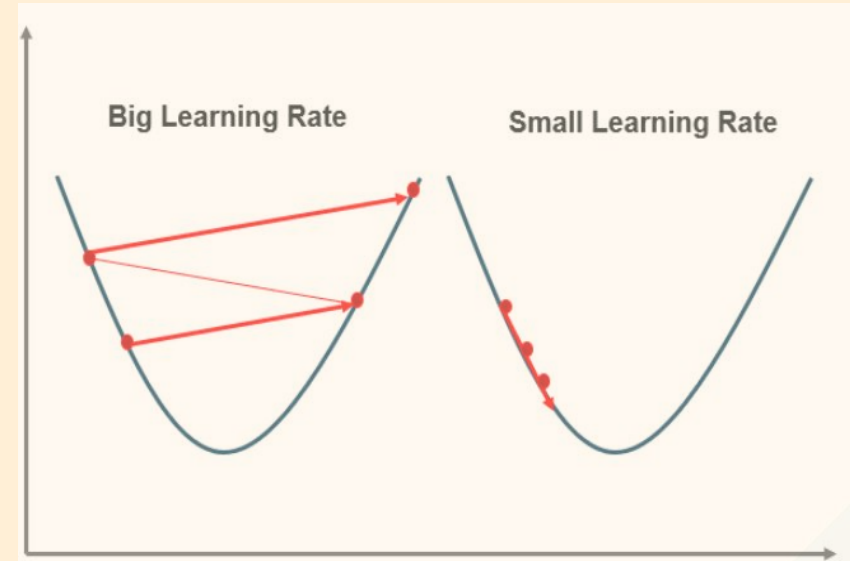
*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Linear regression terminology

- ❖ **Gradient Descent:**
- ❖ It is a method of updating  $b_0$  and  $b_1$  values to reduce the MSE. The idea behind this is to keep iterating the  $b_0$  and  $b_1$  values until we reduce the MSE to the minimum.
- ❖ To update  $b_0$  and  $b_1$ , we take gradients from the cost function. To find these gradients, we take partial derivatives with respect to  $b_0$  and  $b_1$ . These partial derivatives are the gradients and are used to update the values of  $b_0$  and  $b_1$ .



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*



# linear regression - use cases

❖ Sales forecasting

❖ Risk analysis

❖ House price prediction app

❖ Financial applications



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

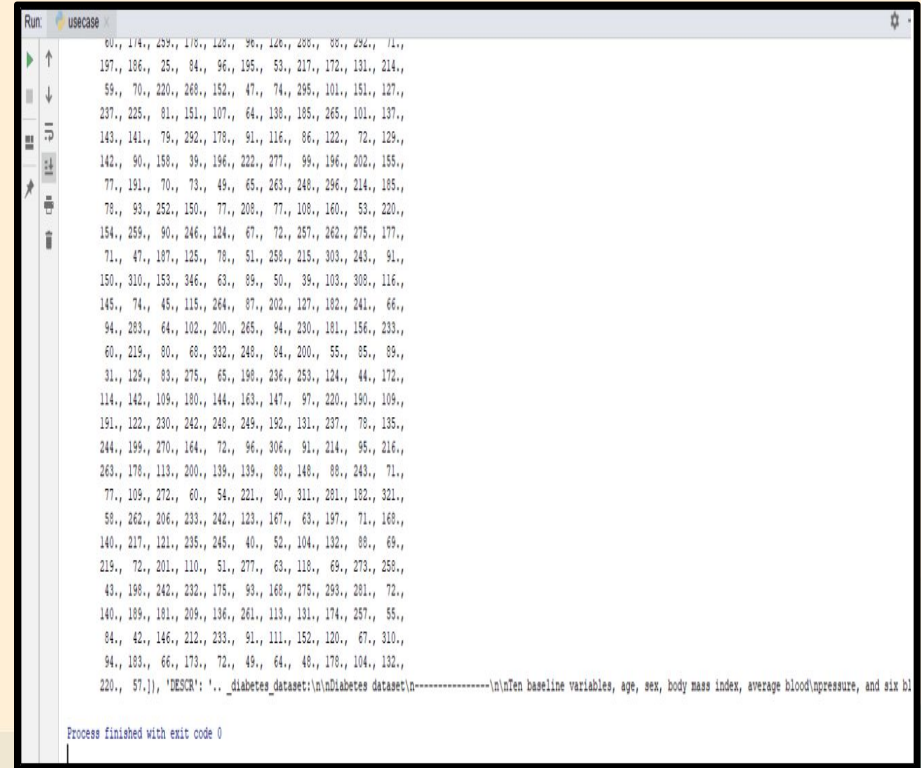
# Simple linear regression

## 1. Loading The Data

We can start with the basic diabetes data set that is already present in the sklearn(scikit-learn) data sets module to begin our journey with linear regression.

```
from sklearn import datasets
```

```
disease = datasets.load_diabetes()  
print(disease)
```



```
Run: usecase  
61., 174., 259., 178., 120., 96., 126., 206., 85., 492., 71.,  
197., 186., 25., 84., 96., 195., 53., 217., 172., 131., 214.,  
59., 70., 220., 268., 152., 47., 74., 295., 101., 151., 127.,  
237., 225., 81., 151., 107., 64., 138., 185., 265., 101., 137.,  
143., 141., 79., 292., 178., 91., 116., 86., 122., 72., 129.,  
142., 90., 158., 39., 196., 222., 277., 99., 196., 202., 155.,  
77., 191., 70., 73., 49., 65., 263., 248., 296., 214., 185.,  
78., 93., 252., 150., 77., 208., 77., 108., 160., 53., 220.,  
154., 259., 90., 246., 124., 67., 72., 257., 262., 275., 177.,  
71., 47., 197., 125., 78., 51., 258., 215., 303., 243., 91.,  
150., 310., 153., 346., 63., 89., 50., 39., 103., 308., 116.,  
145., 74., 45., 115., 264., 87., 202., 127., 182., 241., 66.,  
94., 283., 64., 102., 200., 265., 94., 230., 181., 156., 233.,  
60., 219., 80., 68., 332., 248., 84., 200., 55., 85., 85.,  
31., 129., 83., 275., 65., 198., 236., 253., 124., 44., 172.,  
114., 142., 109., 180., 144., 163., 147., 97., 220., 190., 109.,  
191., 122., 230., 242., 248., 249., 192., 131., 237., 78., 135.,  
244., 199., 270., 164., 72., 96., 306., 91., 214., 95., 216.,  
263., 178., 113., 200., 139., 139., 88., 148., 88., 243., 71.,  
77., 109., 272., 60., 54., 221., 90., 311., 281., 182., 321.,  
58., 262., 206., 233., 242., 123., 167., 63., 197., 71., 168.,  
140., 217., 121., 235., 245., 40., 52., 104., 132., 88., 69.,  
219., 72., 201., 110., 51., 277., 63., 118., 69., 273., 258.,  
43., 198., 242., 232., 175., 93., 168., 276., 293., 281., 72.,  
140., 189., 181., 209., 136., 261., 113., 131., 174., 257., 55.,  
84., 42., 146., 212., 233., 91., 111., 152., 120., 67., 310.,  
94., 183., 66., 173., 72., 49., 64., 48., 178., 104., 132.,  
220., 57.]], 'DESCR': '.._diabetes_dataset:\n\nDiabetes dataset\n-----\n\nTen baseline variables, age, sex, body mass index, average blood pressure, and six bl  
220., 57.]], 'DESCR': '.._diabetes_dataset:\n\nDiabetes dataset\n-----\n\nTen baseline variables, age, sex, body mass index, average blood pressure, and six bl  
Process finished with exit code 0
```



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



trupesh1991@gmail.com

# Simple linear regression

## 2. Exploring The Data

```
print(disease.keys())
```



```
Run: usecase x
C:\Users\Waseem\PycharmProjects\classification\venv\Scripts\python.exe C:/Users/Waseem/PycharmProjects/classification/usecase.py
dict_keys(['data', 'target', 'DESCR', 'feature_names', 'data_filename', 'target_filename'])

Process finished with exit code 0
```

The above code gives all the labels from the data set.



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Simple linear regression

## 2. Exploring The Data

We can slice the data so that we can plot the line in the end. We will also use all the data points, for now, we will slice column 2 from the data.

```
import numpy as np
```

```
disease_X = disease.data[:, np.newaxis,2]
```

```
print(disease_X)
```

```
Run: usecase x
[ 0.07139652]
[-0.02452876]
[-0.0547075 ]
[-0.03638469]
[ 0.0164281 ]
[ 0.07786339]
[-0.03961813]
[ 0.01103904]
[-0.04069594]
[-0.03422907]
[ 0.00564998]
[ 0.08864151]
[-0.03315126]
[-0.05686312]
[-0.03099563]
[ 0.05522933]
[-0.06009656]
[ 0.00133873]
[-0.02345095]
[-0.07410811]
[ 0.01966154]
[-0.01590626]
[-0.01590626]
[ 0.03906215]
[-0.0730303 ]

Process finished with exit code 0
```



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Simple linear regression

## 3. Splitting the Data

We split the data into train and test set.

```
disease_X_train = disease_X[:-30]
```

```
disease_X_test = disease_X[-20:]
```

```
disease_Y_train = disease.target[:-30]
```

```
disease_Y_test = disease.target[-20:]
```

## 4. Generating the model

```
from sklearn import linear_model
```

```
reg = linear_model.LinearRegression()  
reg.fit(disease_X_train,disease_Y_train)
```

```
y_predict = reg.predict(disease_X_test)
```



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Simple linear regression

## 5. Evaluation

We use the mean squared error from the scikit-learn.

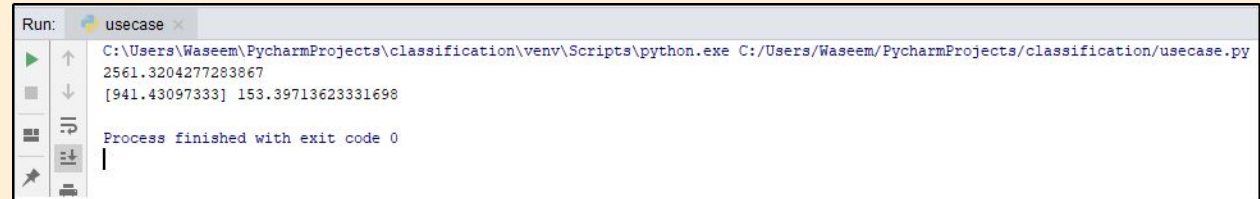
```
accuracy = mean_squared_error(disease_Y_test,y_predict,)
```

```
print(accuracy)
```

```
weights = reg.coef_
```

```
intercept = reg.intercept_
```

```
print(weights,intercept)
```



```
Run: usecase x
C:\Users\Waseem\PycharmProjects\classification\venv\Scripts\python.exe C:/Users/Waseem/PycharmProjects/classification/usecase.py
2561.3204277283867
[941.43097333] 153.39713623331698
Process finished with exit code 0
```



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

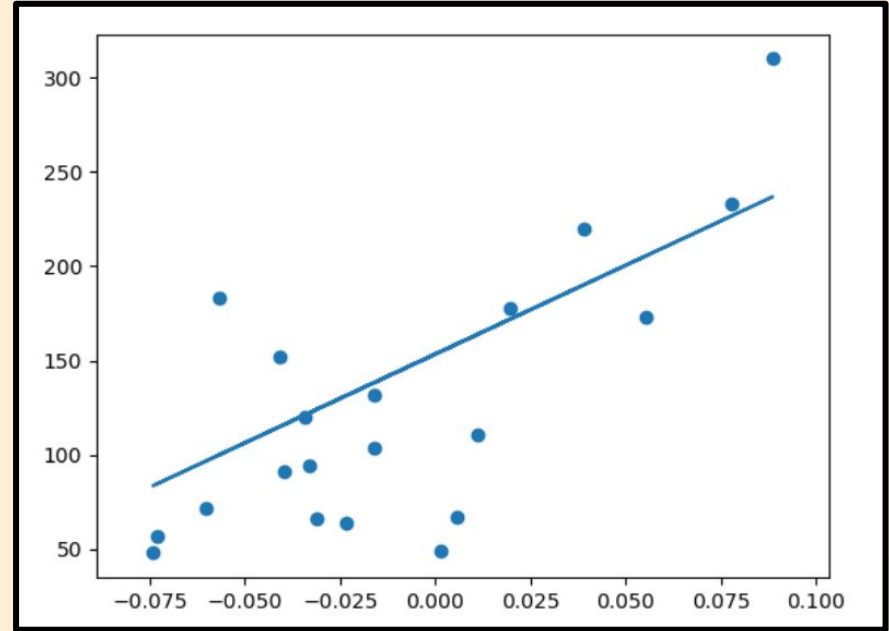
# Simple linear regression

## 5. Evaluation

To be more clear on how the data points look like on the graph, let us plot the graphs as well.

```
import matplotlib.pyplot as plt
```

```
plt.scatter(disease_X_test, disease_Y_test)  
plt.plot(disease_X_test, y_predict)  
plt.show()
```



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



[trupesh1991@gmail.com](mailto:trupesh1991@gmail.com)

# Multiple linear regression

Regression models are used to describe relationships between variables by fitting a line to the observed data.

Regression allows you to estimate how a dependent variable changes as the independent variable(s) change.

Multiple linear regression is used to estimate the relationship between two or more independent variables and one dependent variable.



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*



# Multiple linear regression

You can use multiple linear regression when you want to know:

How strong the relationship is between two or more independent variables and one dependent variable (e.g. how rainfall, temperature, and amount of fertilizer added affect crop growth).

The value of the dependent variable at a certain value of the independent variables (e.g. the expected yield of a crop at certain levels of rainfall, temperature, and fertilizer addition).

You are a public health researcher interested in social factors that influence heart disease. You survey 500 towns and gather data on the percentage of people in each town who smoke, the percentage of people in each town who bike to work, and the percentage of people in each town who have heart disease.

Because you have two independent variables and one dependent variable, and all your variables are quantitative, you can use multiple linear regression to analyze the relationship between them.



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Multiple linear regression

formula:  $y = \beta_0 + \beta_1 X_1 + \dots + \beta_n X_n + \epsilon$

- $y$  = the predicted value of the dependent variable
- $B_0$  = the y-intercept (value of  $y$  when all other parameters are set to 0)
- $B_1 X_1$  = the regression coefficient ( $B_1$ ) of the first independent variable ( $X_1$ ) (a.k.a. the effect that increasing the value of the independent variable has on the predicted  $y$  value)
- ... = do the same for however many independent variables you are testing
- $B_n X_n$  = the regression coefficient of the last independent variable
- $\epsilon$  = model error (a.k.a. how much variation there is in our estimate of  $y$ )



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Multiple linear regression

To find the best-fit line for each independent variable, multiple linear regression calculates three things:

- The regression coefficients that lead to the smallest overall model error.
- The  $t$  statistic of the overall model.
- The associated  $p$  value (how likely it is that the  $t$  statistic would have occurred by chance if the null hypothesis of no relationship between the independent and dependent variables was true).

It then calculates the  $t$  statistic and  $p$  value for each regression coefficient in the model.



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Multiple linear regression

We can predict the CO2 emission of a car based on the size of the engine, but with multiple regression we can throw in more variables, like the weight of the car, to make the prediction more accurate.

Car	Model	Volume	Weight	CO2
Toyota	Aygo	1000	790	99
Mitsubishi	Space Star	1200	1160	95
Skoda	Citigo	1000	929	95
Fiat	500	900	865	90
Mini	Cooper	1500	1140	105
VW	Up!	1000	929	105



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Multiple linear regression

```
import pandas
from sklearn import linear_model

df = pandas.read_csv("data.csv")

X = df[['Weight', 'Volume']]
y = df['CO2']

regr =
linear_model.LinearRegression()
regr.fit(X, y)

#predict the CO2 emission of a car
where the weight is 2300kg, and the
volume is 1300cm3:
predictedCO2 = regr.predict([[2300,
1300]])

print(predictedCO2)
```

## Result:

[107.2087328]

We have predicted that a car with a 1.3 liter engine, and a weight of 2300 kg, will release approximately 107 grams of CO2 for every kilometer it drives.



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Multiple linear regression

## Coefficient

The coefficient is a factor that describes the relationship with an unknown variable.

Example: if  $x$  is a variable, then  $2x$  is  $x$  two times.  $x$  is the unknown variable, and the number  $2$  is the coefficient.

In this case, we can ask for the coefficient value of weight against CO2, and for volume against CO2. The answer(s) we get tells us what would happen if we increase, or decrease, one of the independent values.

```
import pandas
```

```
from sklearn import linear_model
```

```
df = pandas.read_csv("data.csv")
```

```
X = df[['Weight', 'Volume']]
```

```
y = df['CO2']
```

```
regr = linear_model.LinearRegression()
```

```
regr.fit(X, y)
```

```
print(regr.coef_)
```

**Result: [0.00755095 0.00780526]**



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*

# Multiple linear regression

```
import pandas
```

```
from sklearn import linear_model
```

```
df = pandas.read_csv("data.csv")
```

```
X = df[['Weight', 'Volume']]
```

```
y = df['CO2']
```

```
regr = linear_model.LinearRegression()
```

```
regr.fit(X, y)
```

```
print(regr.coef_) Result: [0.00755095 0.00780526]
```

## Result Explained

The result array represents the coefficient values of weight and volume.

Weight: 0.00755095

Volume: 0.00780526

These values tell us that if the weight increase by 1kg, the CO2 emission increases by 0.00755095g.

And if the engine size (Volume) increases by 1 cm3, the CO2 emission increases by 0.00780526 g.



**Prof. Trupesh Patel**

*Assistant Professor, Computer Engineering Department*



*trupesh1991@gmail.com*