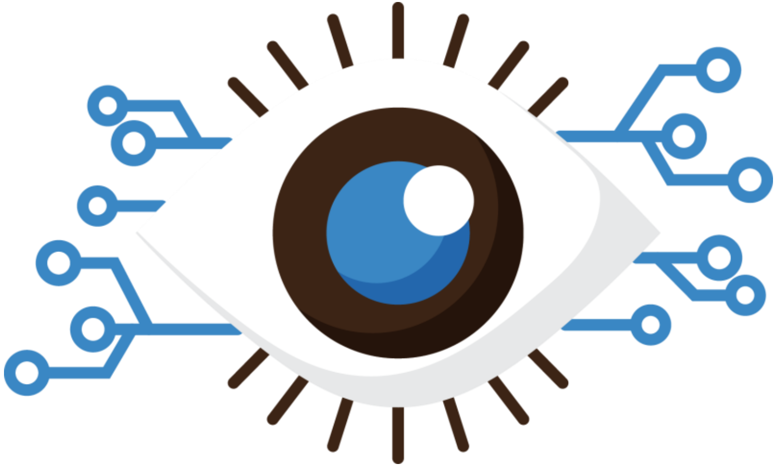
Machine Learning Beginner Course

3.Supervised Learning
-1-

3. التعليم المُراقب
-1-

دورة تعليم الالة للمبتدئين



Salam again!

In the past classes did a great overview of machine learning, know it’s aspects and prepare the data in the preprocessing phase, today we will kick off the Supervised Type of Machine Learning.

It's hardware that makes a machine fast. It's software that makes a fast machine slow.

Supervised Learning

Its about feeding input and the desired output to the ML algorithm which will find some correlation between the input and the output in order to use it later to do prediction.

$$X \xrightarrow[f(x)=y]{} Y$$

Example

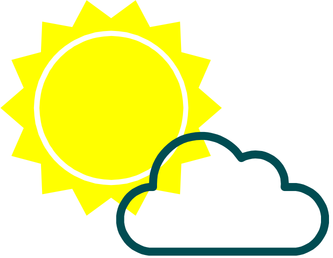
Country	Age	Salary	Purchased
Algeria	44	72000	0
Tunis	27	48000	1
Morocco	30	54000	0
Tunis	38	61000	0
Morocco	40	60333.33	1
Algeria	35	58000	1
Tunis	37.333	52000	0
Algeria	48	79000	1
Morocco	50	83000	0
Algeria	37	67000	1

X Y

Type of Output

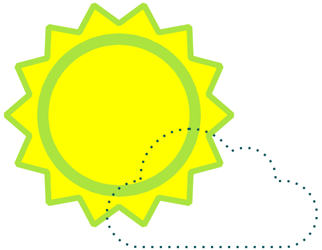
The output can be in two main type, either discrete or continuous, if it is the discrete case we say it’s a problem of **classification** otherwise it’s a problem of **Regression**

Example



Regression
What is the temperature going to be tomorrow ?

38 °



Classification
Will it be Cold or Hot tomorrow?

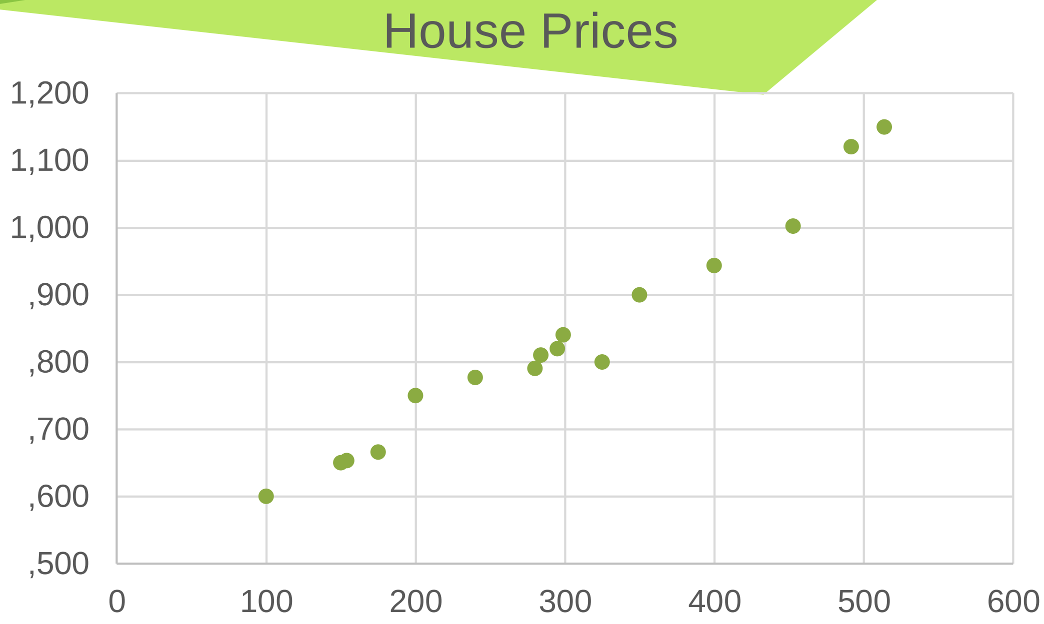
Cold

1.

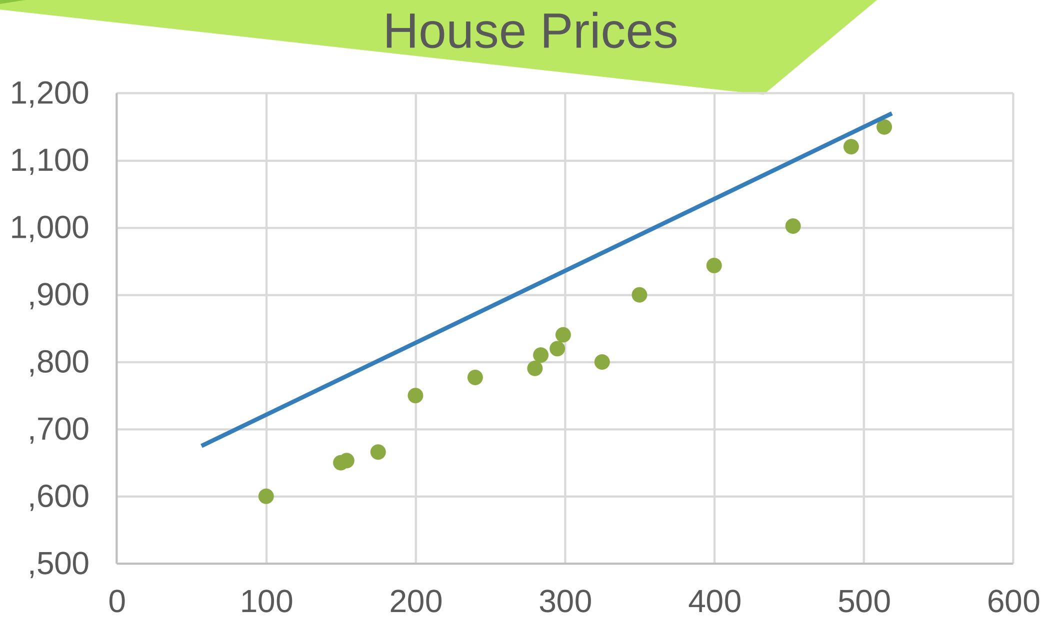
Regression

The Continuous type of the Output.

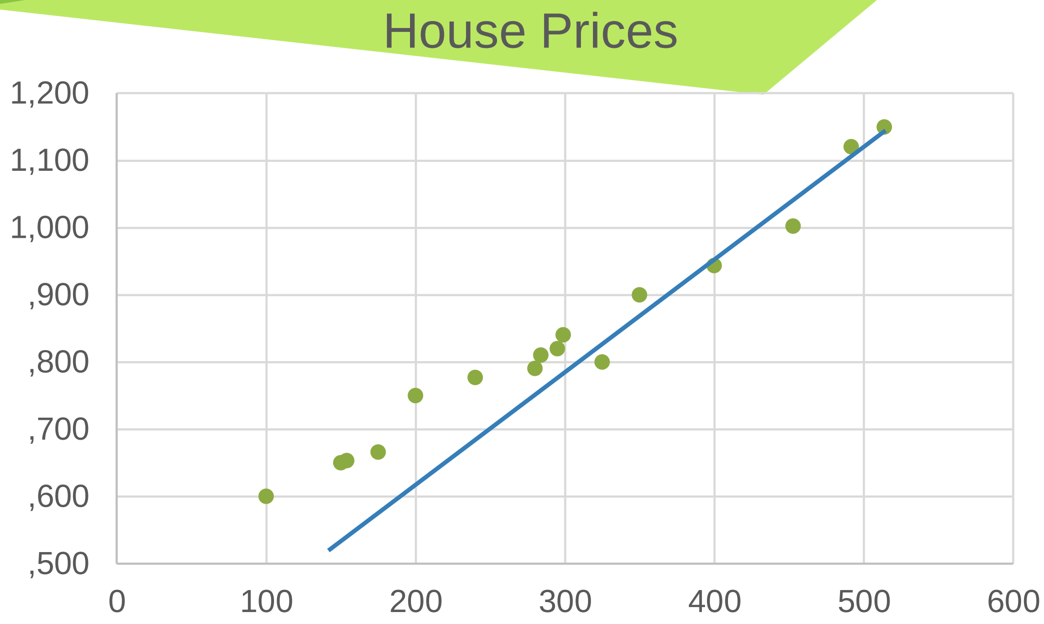
Famous Example



Is This line Good ?



And This?


House Prices
1,200
1,100
1,000
,900
,800
,700
,600
,500
0
100
200
300
400
500
600

And Now!

House Prices
1,200
1,100
1,000
,900
,800
,700
,600
,500
0
100
200
300
400
500
600

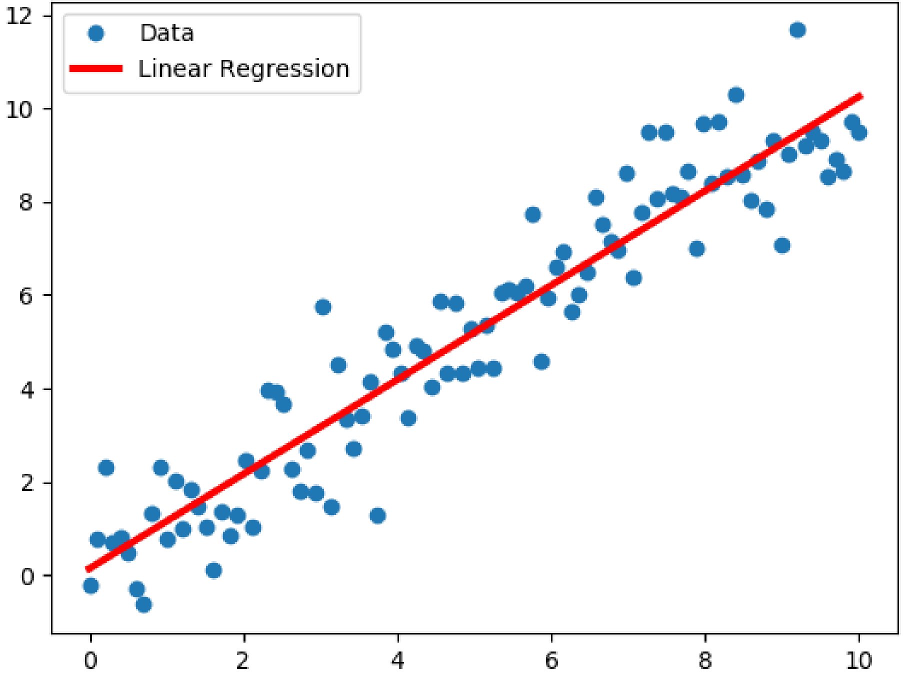
The Math behind

Linear Regression

Simple Linear Regression

1. We are given a set of points : $\{(x_1, y_1),$ $(x_2, y_2),$ $(x_3, y_3),\ldots.,$ $(x_n, y_n)\}$.
2. We plot them in a 2-D Chart.
3. We find the line of best fit.

Simple Linear Regression



معادلة المستقيم

Line Equation

$$y = ax + b$$

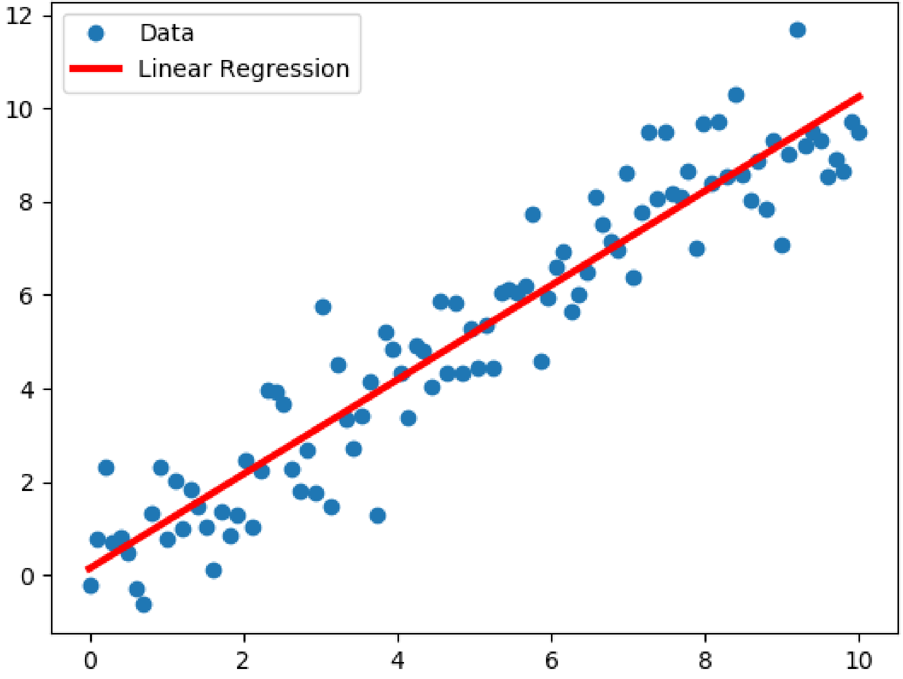
Slope

معامل التوجيه

Intercept

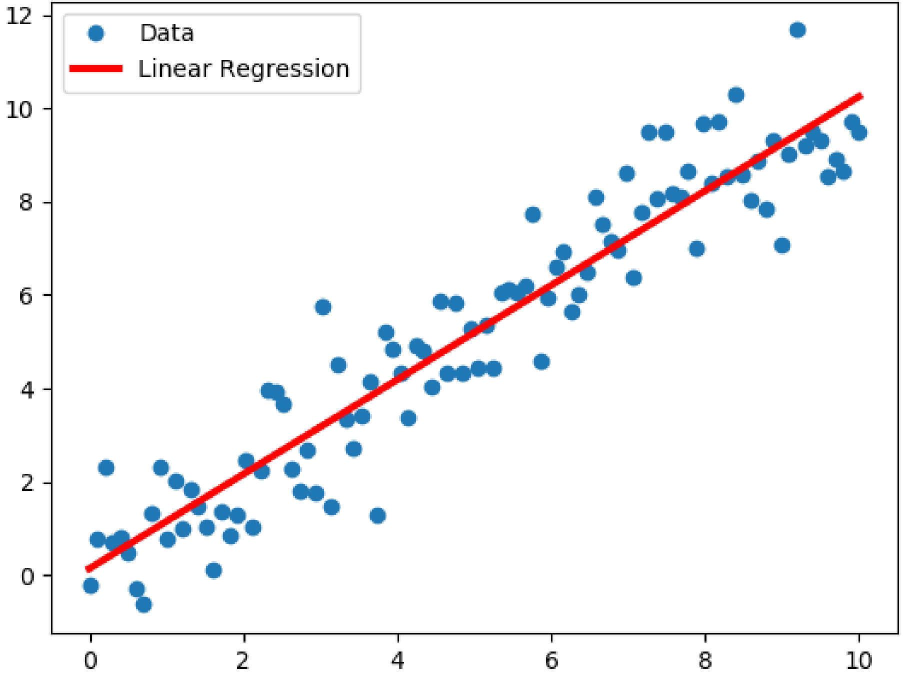
الارتوب

Cost Function



A cost function (objective or error function) is a function that tells us how good we are in the way of finding the best **a** and **b**

Cost Function



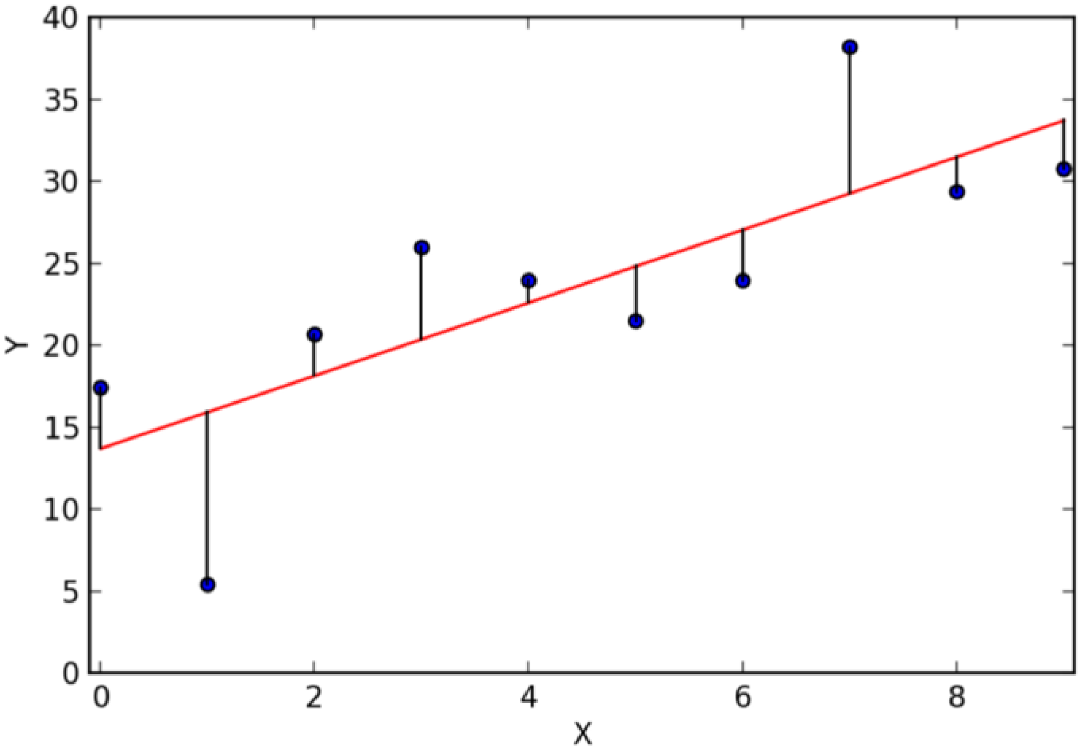
$$\hat{y}_i = a x_i + b$$

We would like:

$\hat{y}_i$ close to y_i

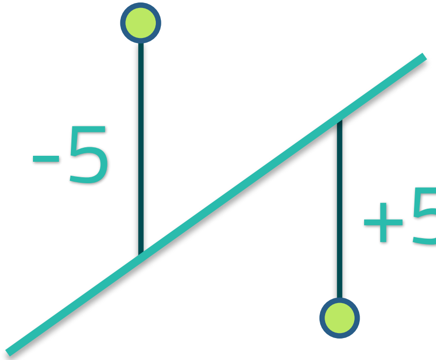
for i = 1..N

Errors on Each Point



Can we Do this?

$$Error\,? = \sum_{i=1}^{N} (y_i - \hat{y_i})$$



If one sample gives us +5, and other -5 the sum would b 0! And this is clearly not 0 error.

Simple Linear Regression

We want a positive contribution to error.

$$|x|$$

القيمة المطلقة

أو

$$x^2$$

التربيع

التربيع

$$E = \sum_{i=1}^{N} (y_i - \widehat{y_i})^2$$

Sum of Squared Error

Minimize Cost Function

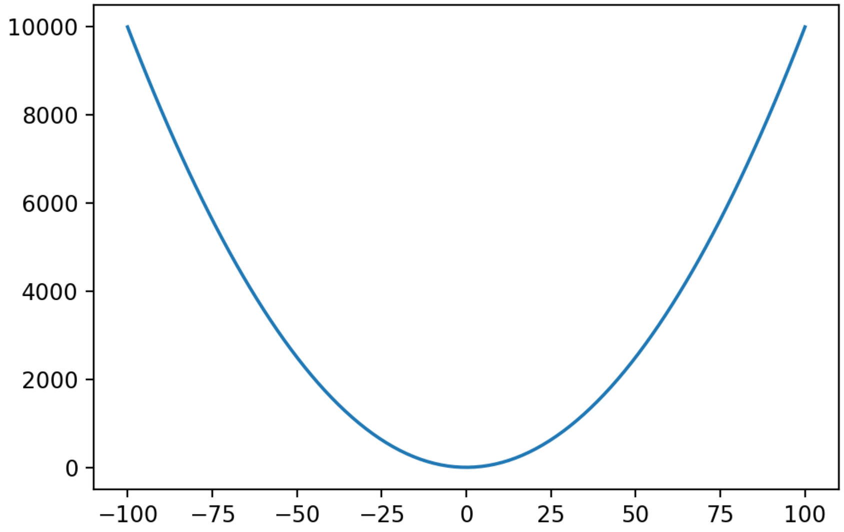
Our Friend Calculus Again

$$x^2$$

Has a Lower Bound

Which correspond to the

best **a** and **b**



Minimize Cost Function

So Here is again Our Cost Function

$$E = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

Substitute with our expression

$$E = \sum_{i=1}^{N} (y_i - (ax_i + b))^2$$

Remember that y_i and x_i are given and we want to find the **a** and the **b**

Minimize Cost Function

We do have a function with two variables a and b

$$E = \sum_{i=1}^{N} (y_i - (ax_i + b))^2$$

Therefore we need to use partial derivatives

$$\partial$$

It’s derivative of a function of two or more variables with respect to one variable, the other(s) being treated as **constant**.

Example

$$f(x, y) = 2x^2 + y^2$$

$$\frac{\partial f}{\partial y} = 2y$$

$$\frac{\partial f}{\partial x} = 4x$$

$$g(x, y) = e^{xy}$$

$$\frac{\partial g}{\partial y} = xe^{xy}$$

$$\frac{\partial g}{\partial x} = ye^{xy}$$

Minimize Cost Function For a

$$E = \sum_{i=1}^{N} (y_i - (ax_i + b))^2 \qquad (f^n)' = nf'f^{n-1}$$

$$\frac{\partial E}{\partial a} = \sum_{i=1}^{N} 2\left(y_i - (ax_i + b)\right)(-x_i)$$

Set it to Zero

$$\sum_{i=1}^{N} 2\left(y_i - (ax_i + b)\right)(-x_i) = 0$$

$$-\sum_{i=1}^{N} y_i x_i + a\sum_{i=1}^{N} {x_i}^2 + b\sum_{i=1}^{N} x_i = 0$$

Set it to Zero

$$a\sum_{i=1}^{N} x_i{}^2 + b\sum_{i=1}^{N} x_i = \sum_{i=1}^{N} y_i\, x_i$$

Make every thing positive
We do have 2 unknowns and One Equation, therefore
we still need to take the derivative of **E** WRT **b**

Minimize Cost Function For a

$$\frac{\partial E}{\partial b} = \sum_{i=1}^{N} 2\big(y_i \; - (ax_i \; + b)\big)(-1)$$

$$-\sum_{i=1}^{N} y_i + a\sum_{i=1}^{N} x_i + b\sum_{i=1}^{N} 1 = 0$$

$$a\sum_{i=1}^{N} x_i + bN = \sum_{i=1}^{N} y_i$$

Solve System of Equation

Recall from middle school math how to solve2 equation and 2 unknowns

حل جملة معادلتين

$$a\sum_{i=1}^{N} {x_i}^2 + b\sum_{i=1}^{N} x_i = \sum_{i=1}^{N} y_i\, x_i$$

$$a\sum_{i=1}^{N} x_i + bN = \sum_{i=1}^{N} y_i$$

Replacing

Lets Replace summation by letters

$$C = \sum_{i=1}^{N} {x_i}^2, \qquad D = \sum_{i=1}^{N} x_i, \qquad E = \sum_{i=1}^{N} y_i\, x_i\ , \qquad F = \sum_{i=1}^{N} y_i$$

$$a\sum_{i=1}^{N} {x_i}^2 + b\sum_{i=1}^{N} x_i = \sum_{i=1}^{N} y_i\, x_i$$

$$a\sum_{i=1}^{N} x_i + bN = \sum_{i=1}^{N} y_i$$

$$\Rightarrow$$

$$aC + bD = E$$

$$aD + bN = F$$

Solving For a

$$(aC + bD = E) \times N$$

$$(aD + bN = F) \times D$$

$$aCN + bDN = EN \ldots.(1)$$

$$aD^2 + bDN = FD \ldots.(2)$$

Subtract 1 from 2 and we get:

$$a(CN - D^2) = EN - FD$$

$$a = \frac{EN - FD}{CN - D^2}$$

Solving For b

$$(aC + bD = E) \times D \quad \Rightarrow \quad aCD + bD^2 = ED \;.... (1)$$

$$(aD + bN = F) \times C \quad \Rightarrow \quad aCD + bCN = FC \;.... (2)$$

Subtract 1 from 2 and we get: $b(D^2 - CN) = ED - FC$

$$b = \frac{ED - FC}{D^2 - CN}$$

Plug Back

$$a = \frac{EN - FD}{CN - D^2} \Rightarrow a = \frac{N\sum_{i=1}^{N} y_i\, x_i - \sum_{i=1}^{N} y_i \sum_{i=1}^{N} x_i}{N\sum_{i=1}^{N} {x_i}^2 - \left(\sum_{i=1}^{N} x_i\right)^2}$$

$$b = \frac{ED - FC}{D^2 - CN} \Rightarrow b = \frac{\sum_{i=1}^{N} y_i\, x_i \sum_{i=1}^{N} x_i - \sum_{i=1}^{N} y_i \sum_{i=1}^{N} {x_i}^2}{\left(\sum_{i=1}^{N} x_i\right)^2 - N\sum_{i=1}^{N} {x_i}^2}$$

Simplify and Final Solution

$$a = \frac{N \sum_{i=1}^{N} y_i\, x_i - \sum_{i=1}^{N} y_i \sum_{i=1}^{N} x_i}{N \sum_{i=1}^{N} {x_i}^2 - \left(\sum_{i=1}^{N} x_i\right)^2}$$

$$b = \frac{\sum_{i=1}^{N} y_i \sum_{i=1}^{N} {x_i}^2 - \sum_{i=1}^{N} y_i\, x_i \sum_{i=1}^{N} x_i}{N \sum_{i=1}^{N} {x_i}^2 - \left(\sum_{i=1}^{N} x_i\right)^2}$$

Numpy Trick

Dot Product $$a^T b = \sum_{i=1}^{N} a_i \, b_i$$

Numpy also have mean() and sum() function that we'll find useful

Never do this

```
total = 0
for i in range(N):
    total += a[i] * b[i]
```

Instead

```
total = a.dot(b)
total = np.dot(a, b)
```

Sklearn Simplification

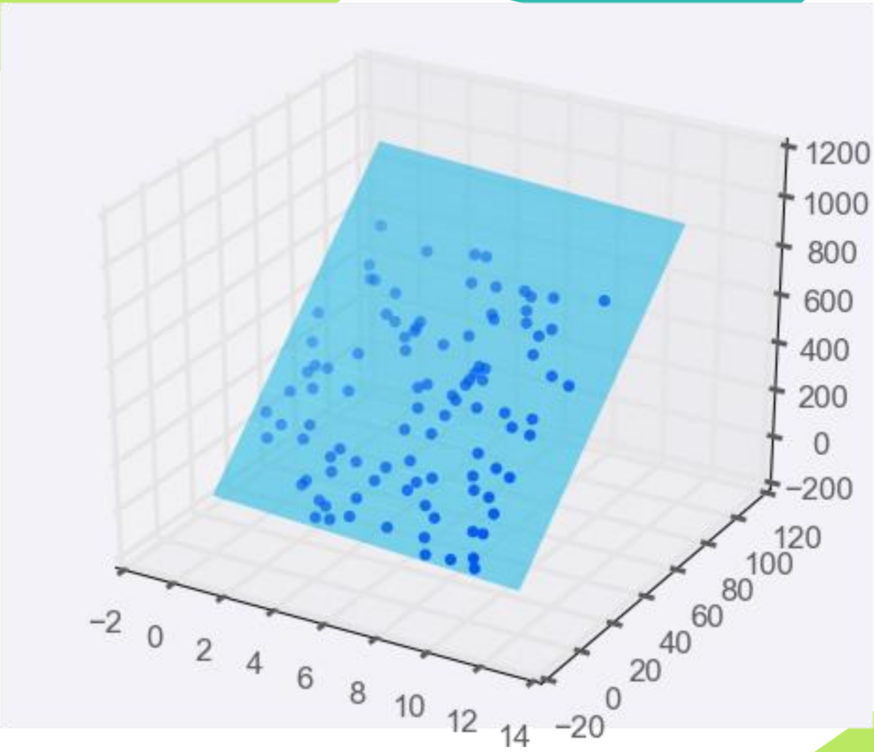
```
import pandas as pd

df = pd.read_csv(“somedata.csv")

X = df.iloc[:,1:2].values

y =df.iloc[:,2].values

from sklearn.linear_model import LinearRegression

regressor = LinearRegression()

regressor.fit(X, y)
```

Multiple Linear Regression

Complex Problem

We use multiple Regression when there are more than one set of input features as the equation states :

$$y = a_1x_1 + a_2x_2 + \cdots + a_nx_n$$



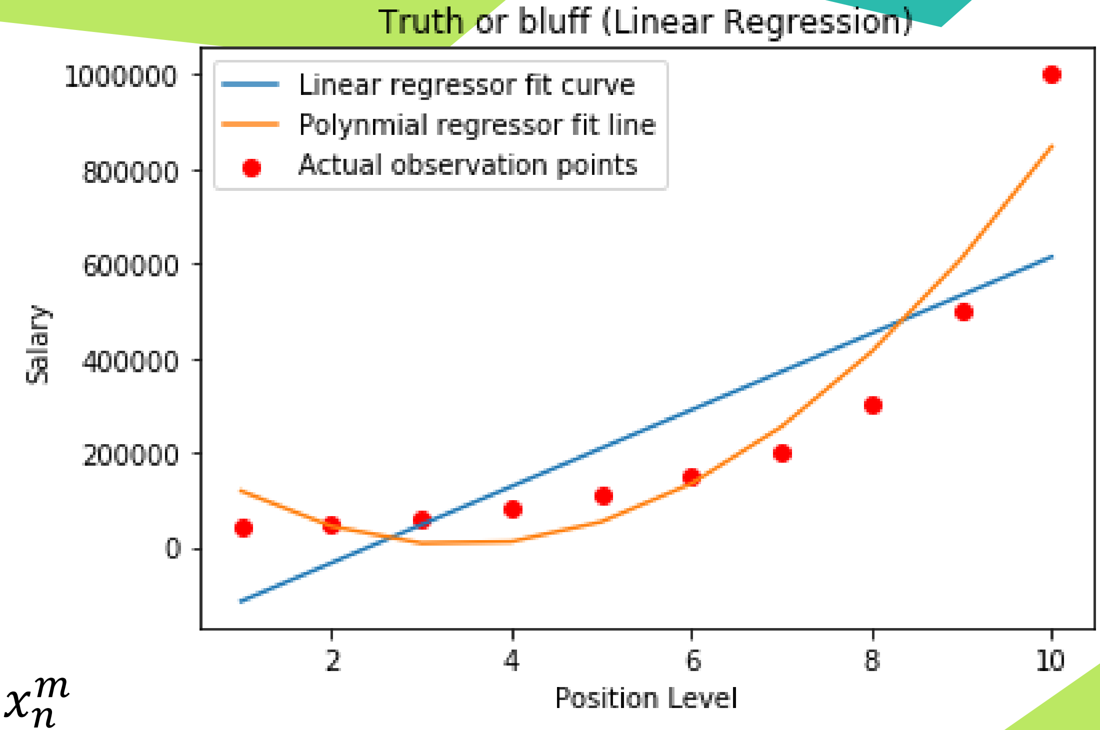
Sklearn Again

```
import pandas as pd

df = pd.read_csv(“somedata.csv")

X = df.iloc[:, :-1].values

y =df.iloc[:, -1].values

from sklearn.linear_model import LinearRegression

regressor = LinearRegression()

regressor.fit(X, y)

regressor.predict(somevalue)
```

Polynomial Linear Regression

Complex Problem Again

We use Equation with polynomials, the relationship between the independent variable x and the dependent variable y is modelled as an nth degree polynomial in x



$$y = a_1 x_1 + \mathrm{a}_2 {x_1}^2 + \cdots + a_\mathrm{n} x_n^m$$

Again With Sklearn

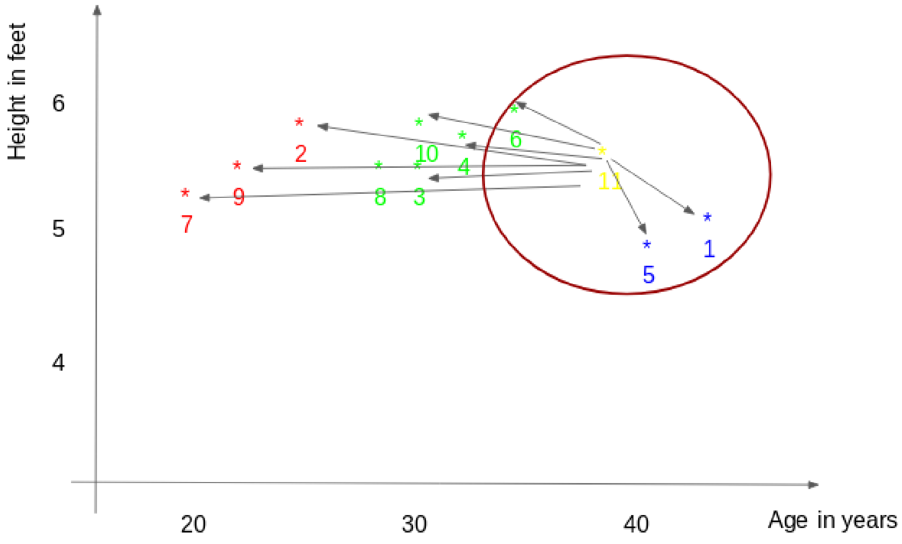
```
import pandas as pd
df = pd.read_csv(“somedata.csv")
X = df.iloc[:, :-1].values
y =df.iloc[:, -1].values
from sklearn.preprocessing import PolynomialFeatures
poly_reg = PolynomialFeatures(degree = 2)
X_poly = poly_reg.fit_transform(X)
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_poly, y)
regressor.predict(somevalue)
```

Other
Regression Algorithm

Other Algorithms

K-Neighbors Regression (KNR)

https://www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python/



[*https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html*](https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html)

Other Algorithms

Random Forest Regression (RFR)

https://medium.com/datadriveninvestor/random-forest-regression-9871bc9a25eb

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

Decision Tree Regression (DTR)

https://medium.com/data-py-blog/decision-tree-regression-in-python-b185a3c63f2b

https://scikit-learn.org/stable/auto_examples/tree/plot_tree_regression.html

Other Algorithms

Support Vector Regression (SVR)

https://medium.com/coinmonks/support-vector-regression-or-svr-8eb3acf6d0ff

https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html

Artificial Neural Network (ANN)

https://medium.com/@gautam.karmakar/linear-regression-using-neural-network-d8815324017f

https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html

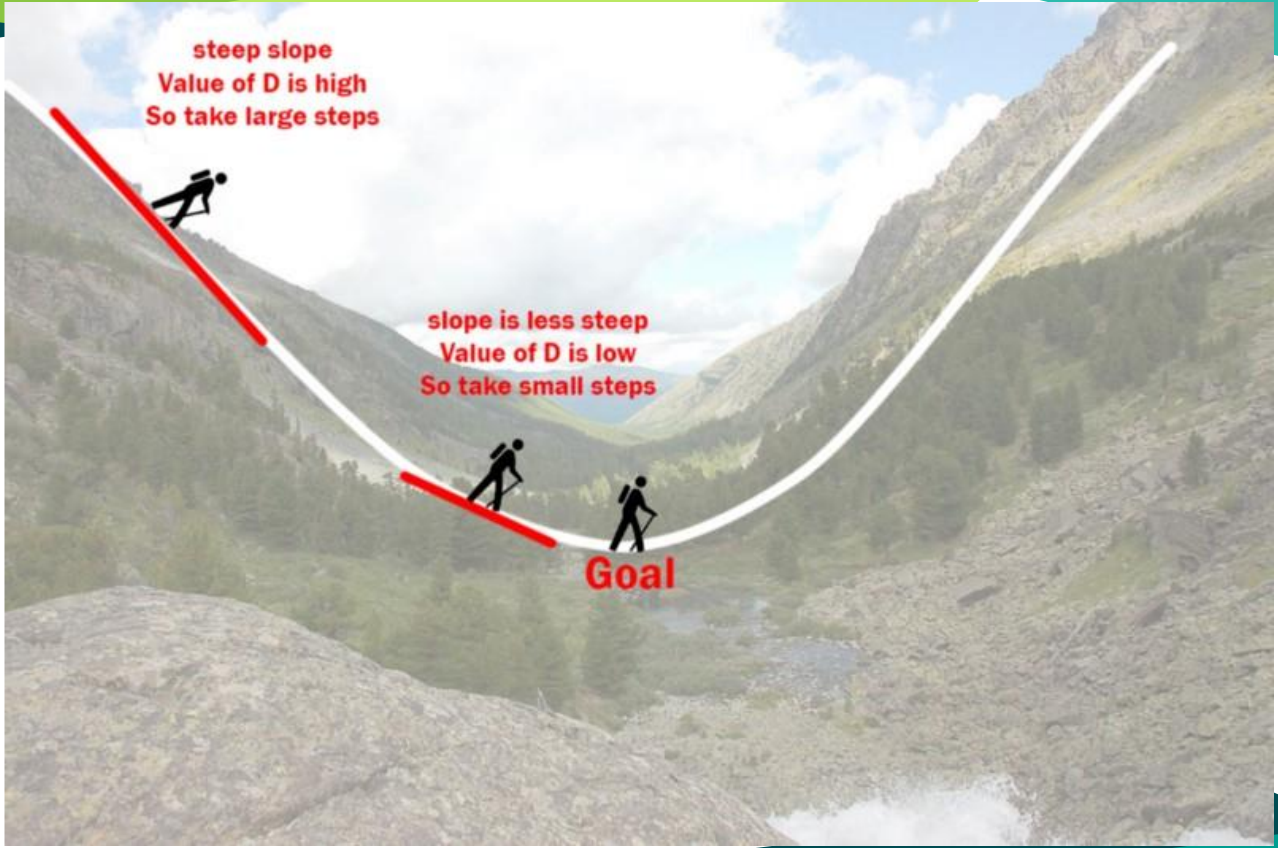
Gradient Descent

A way for optimization.

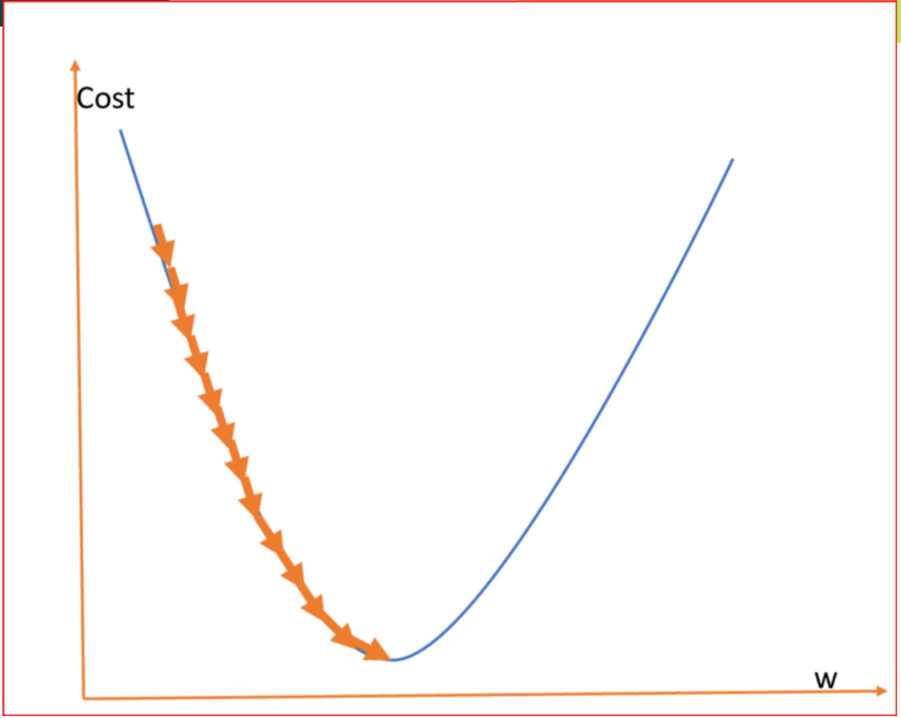
Gradient Descent Algorithm

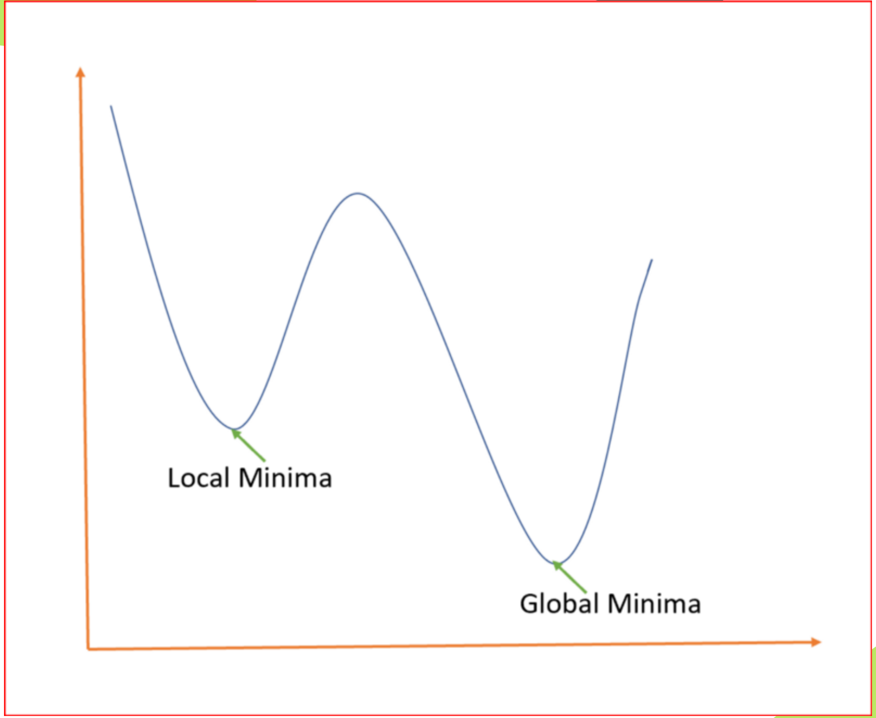
It is an optimization algorithm used in training a model. In simple words, Gradient Descent finds the parameters that minimize the cost function (error in prediction). Gradient Descent does this by iteratively moves toward a set of parameter values that minimize the function, taking steps in the opposite direction of the gradient.

Gradient Descent Algorithm

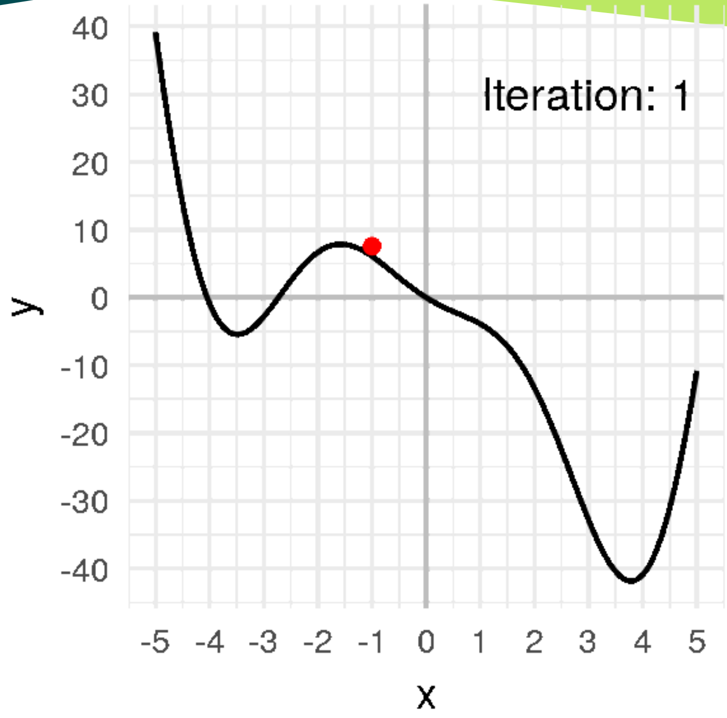


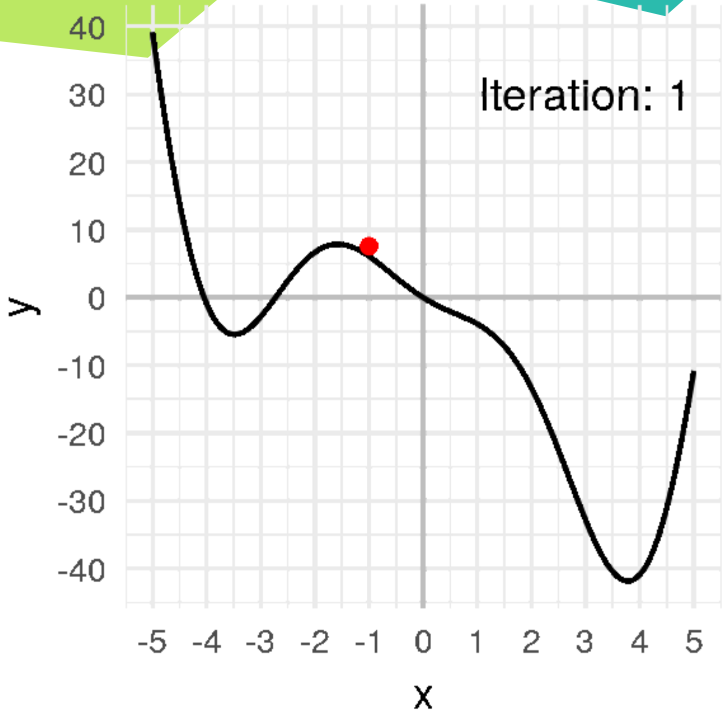
Gradient Descent Algorithm





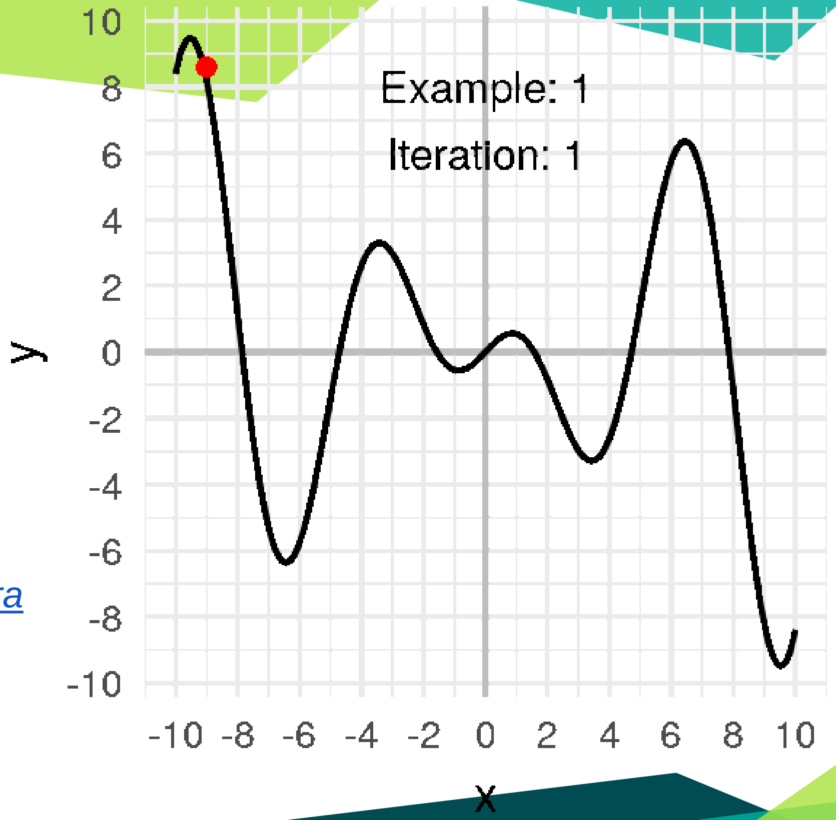
Gradient Descent Algorithm





Local Minima Problem

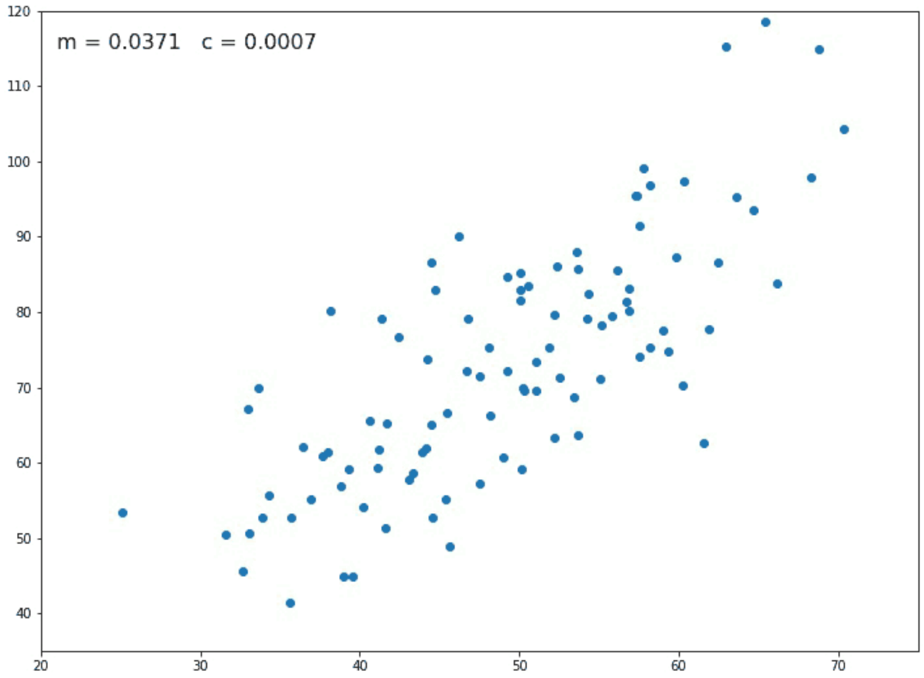
A solution to the local minima problem was another variant called Stochastic gradient descent which start at random different position.



https://medium.com/konvergen/gradient-descent-and-stochastic-gradient-descent-algorithms-for-neural-networks-e817f3c411ef

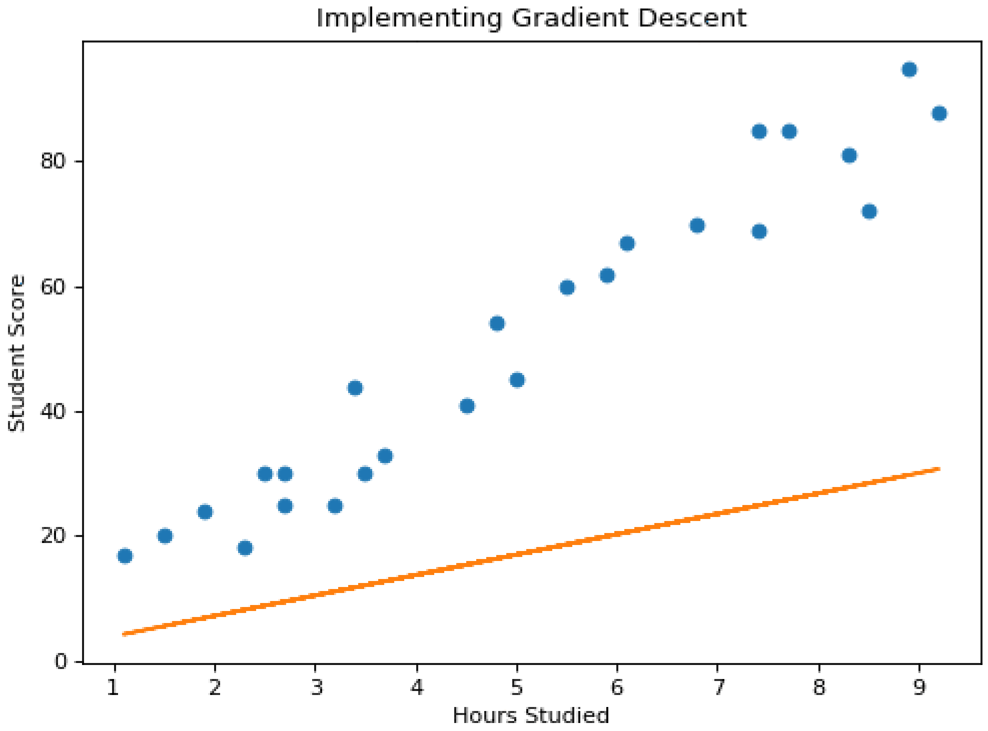
Example In Regression

We can use GD in Linear Regression to find an approximate **a** and **b**



https://towardsdatascience.com/linear-regression-using-gradient-descent-97a6c8700931

Another Example In Regression



Practical Time

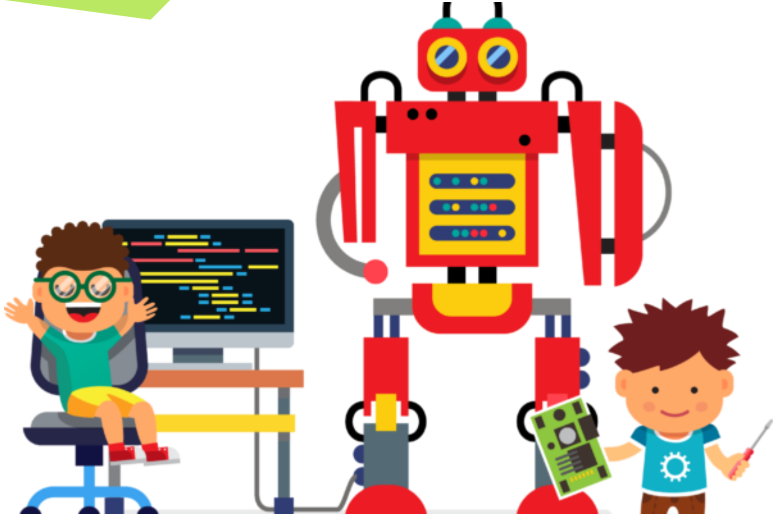
Open up your PC, launch your anaconda and let's create some regression model.

Coding Interview

amazon

Given a sorted array, find the smallest positive integer that is not the sum of a subset of the array.

For example, for the input [1, 2, 3, 10], you should return 7.
Do this in O(N) time.



شكرا لحضوركم

Thanks for Assisting!