Linear Regression

Machine Learning

Machine Learning is Nothing Fancy, Just intelligently used mathematics if you care to dig a little bit deeper.

Task vs Algorithm

We have various tasks to solve and then we have hundred of or even thousand of different algorithms to solve those tasks.

For Eg:
Task = solve a quadratic equation
Algo = We know the discriminant rule

Task = Predict the price of the House
Algo = Linear Regression

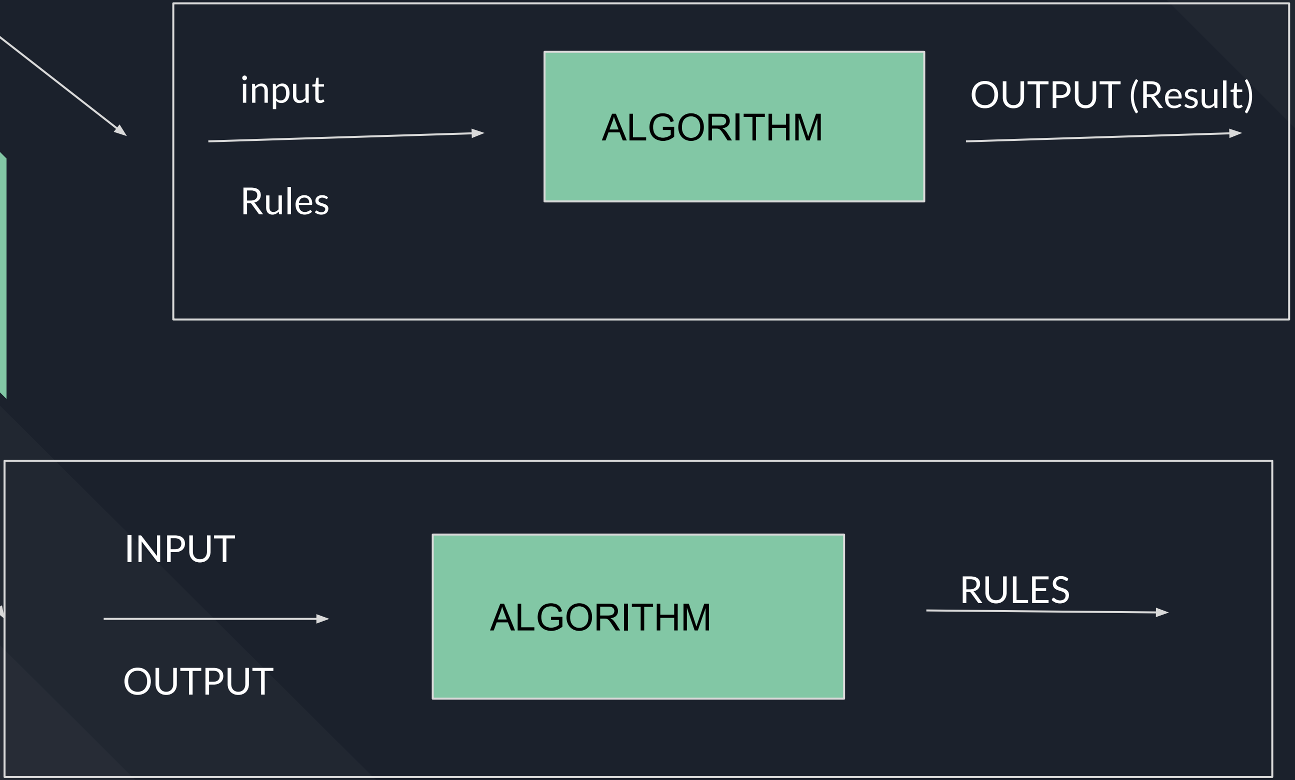
Programmable vs Not Programmable

A task for which the algorithm can be clearly written on a piece of paper can be considered a task which is programmable.

ML is about solving those non programmable tasks

Tasks which are not programmable are generally those tasks which we humans can easily do. These are the tasks which ML tries to solve.

CS ALGORITHM VS ML ALGORITHM



Okay, but what is Linear Regression?

Remember, Task Vs Algorithm.

In Machine Learning, we study various algorithms to solve specific tasks.

Linear Regression is an algorithm.

So, what type of task does it solve?

It solves tasks where the variable that we want to predict is of continuous type.

For Example: Prediction of Price of the House, Height or weight of an individual, revenue in coming year for a firm… etc

These types of tasks are collectively known as “Regression Problem”

What if the type is not continuous?

Then the type most likely will be categorical where the no of distinct values are countable.

For example:

1). Predicting whether a given email is spam or not,
2). Predicting whether a given tweet is positive, negative or neutral,
3). Predicting whether an image has a cat or dog in it.. Etc

These are all classification tasks,

These types of tasks are collectively known as “Classification Problems.”

So, to summarize, we have Regression and classification tasks.
In this video, we are studying one algorithm to solve Regression Task.

Again, there can be many algorithms to solve regression related tasks but we are studying Linear Regression in this Video.

Let’s Study how does Linear Regression Work….

Let us say the variable that we want to predict is Price of House. So, this variable will be our y

Let us suppose as feature, we have information on number of rooms. This will be our X.

We have for the time being assumed only one X but the results are easily extended for more than one X.

So, what is Linear Regression?

Linear Regression is a Procedure to find a Linear expression for a Line between y and X.

Linear Expression in Two Variables

$Y = mx + c$

Where

m = slope of the line

c =   Intercept of the line

Examples of Lines:

$Y = 2x + 10$

$Y = 3x + 23$

$Y = 1.4x + 34$

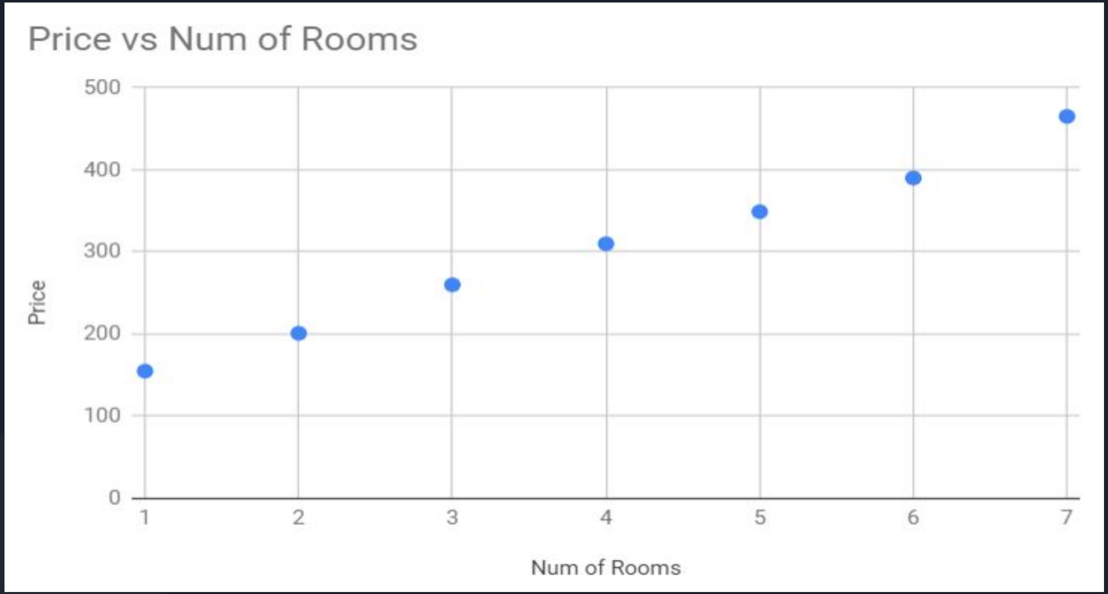
$Y = 0.4x + 36$

So, you see there are many expressions but which one is best?

Best Line for Our Data

Num of Rooms	Price
1	155
2	201
3	260
4	310
5	349
6	390
7	465

Let's Plot the data



We have just seen that the previous plot can be fitted but many lines

But which one is best?

The one…

Which is as close as possible to the data.

So, how to we teach compute to find such a line?

Computers are best when it comes to doing thousands or even lakhs of calculations in seconds..

So, what we do is...

1). Take some random values of m and c.
2). Decide the number of iterations.
3). Repeat till number of iterations:
   a) Take random data (X,y)
   b) Obtain Prediction using the above random values of m and c
   c) Update this values by a very small margin called learning rate.

The idea is that if we do this many times (large number of times), then eventually the values of m and c will be perfect.

The only question is: How do we update the values of m and c?

In each loop, what we do is:

1) Check the Prediction and Check its actual value.
2) Then, if the prediction is close to the actual value, we don’t change much.
3) If it is far away, we move the line towards that actual point by changing its m and c.

Pseudocode For Linear Regression

```
# Step.1 Take some number of iterations...
number_of_iterations = 10000
# Step. 2 Intialize some random values of m and c
m = some_random
c = some_random
# Step. 3 Loop through number_of_iterations times
for value in range(number_of_iterations):
  # Randomly choose a data point (X and y)...
  random_X = something
  random_y = something
  # For each loop we obtain the prediction for the current m and c...
  prediction = m * random_X + x
  # then we compare, prediction with actual y
  # then we adjust m and c accordingly.
  # Usually, we take the difference of prediction and actual y
  # Because if the difference is close to zero, then that means we don't
  # have to Update that much..
  m += (actual_y - prediction) * random_X * learning_rate
  c += (actual_y - prediction) * learning_rate

# Step. 4 After the loop ends, we get a value of m and c which is hopefully
# near to perfect.
```

The difference between actual and Prediction makes sense but why in update of m, we multiple random_x?

The reason is m stands for slope which is dependent on the sign of the input x.

In our case, X is number of rooms which cannot be negative, so no need to worry about that..

I hope you were able to follow through and understand everything that we discussed.

Thank You.