Supervised ML Algorithms - Regression

Evaluating a Regression Model

dφ

Democratizing Data Science Learning

DPhi

Learning Objectives

Cost

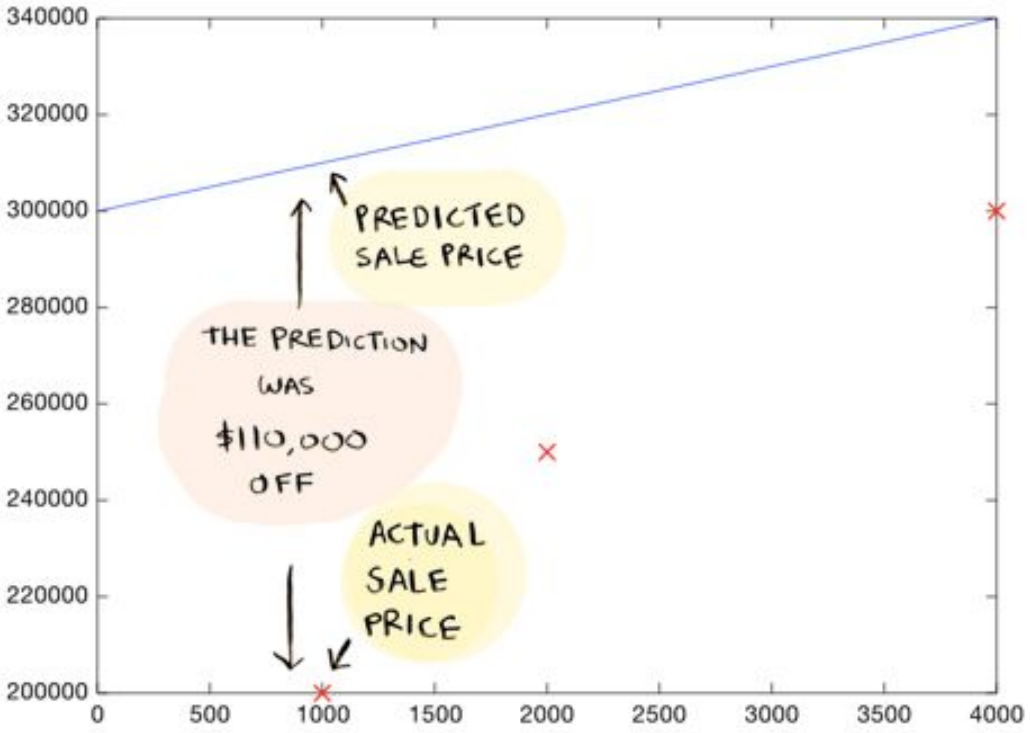
Cost Function

Gradient Descent

DPhi

Which line is good?

Now coming back to our first example. **How do you decide what line is good? Here's a bad line:**

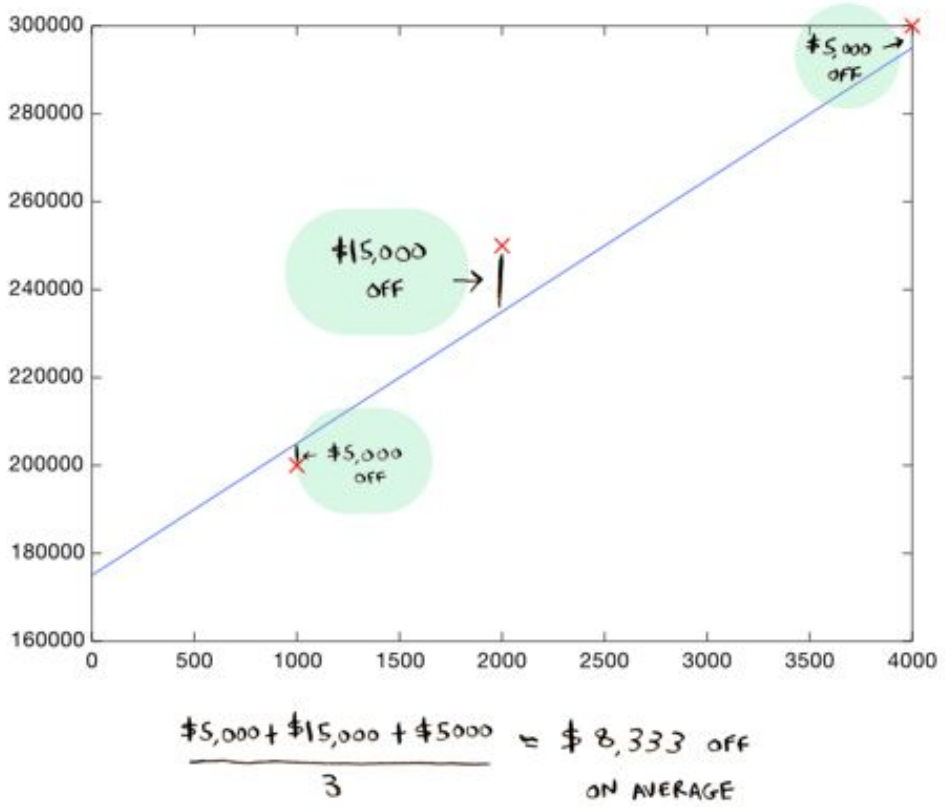


This above drawn line is way off. For example, according to the line, a 1000 sq foot house should sell for $310,000, whereas we know it actually sold for $200,000.

DPhi

Which line is good?

Here's a better line:



This line is an average of $8,333 dollars off (adding all the distances and dividing by 3).

This $8,333 is called the **cost** of using this line.

DPhi

Short-term Objective

What were we doing in the previous 2 examples? We plotted 2 straight lines using the equation: y = mx+b.

If we already have the data points (x1, y1), ..., (xn, yn), it means that our values of x and y remain the same throughout all the lines we plot.

So what remains? What exactly are we changing to plot different lines? Yes, m and b.

Our objective is to find the values of m and b that will best fit this data.

These 2 variables are actually called **hyperparameters.** In machine learning, **a hyperparameter is a parameter whose value is used to control the learning process. And we must always try to find some optimal parameters while building a machine learning model.**

Cost

The **cost** is how far off the line is from the real data. The best line is the one that is the least off from the real data.

To find out what line is the best line (to find the values of m and b), we need to use a **cost function**.

In ML, cost functions are used to estimate how badly models are performing.

Put simply, a cost function is a measure of how wrong the model is in terms of its ability to estimate the relationship between X and y.

DPhi

Cost Function

What?

Now that we built a model, we need to measure its performance right? and understand if it works well or not. Cost function measures the performance of a Machine Learning model for given data. It quantifies the error between predicted values and expected values and presents it in the form of a single real number.

Depending on the problem Cost Function can be formed in many different ways. The purpose of this function is to be either:

- **Minimized** - then returned value is usually called **cost, loss or error.** The goal is to find the values of model parameters for which Cost Function return as small number as possible.
- **Maximized** - then the value it yields is named a **reward**. The goal is to find values of model parameters for which returned number is as large as possible.

What is predicted and expected value?

- **Predicted value:** As the name says is the predicted value of your machine learning model.

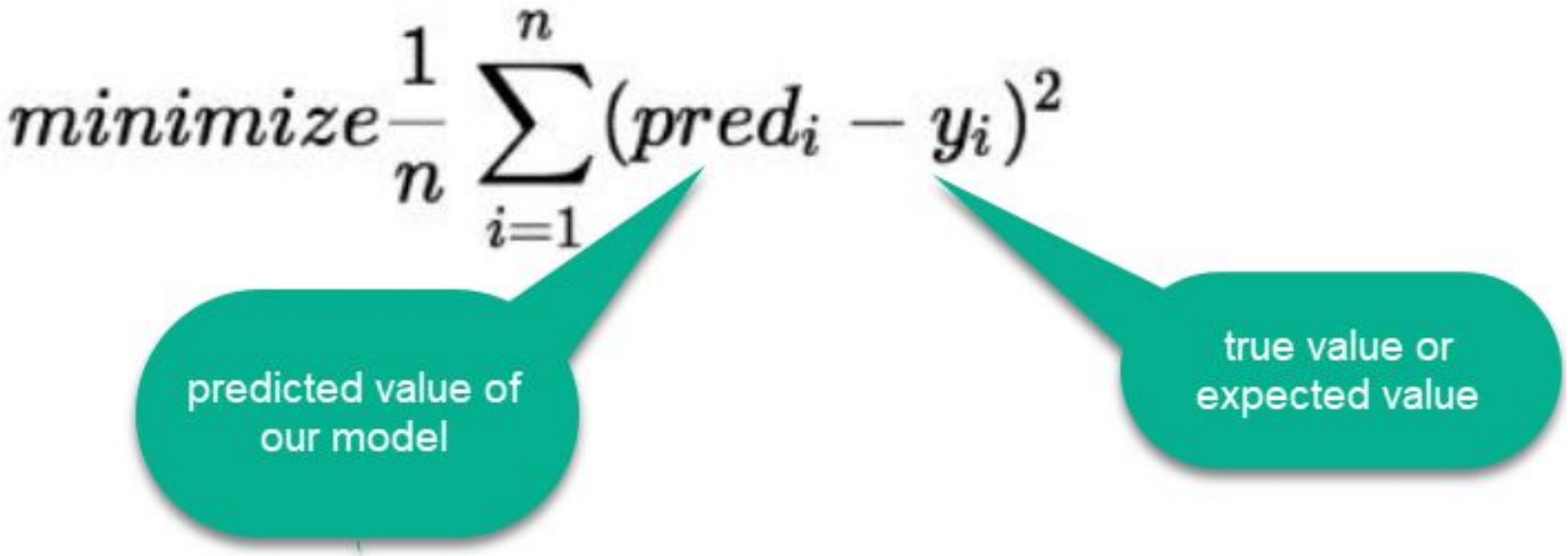
- **Expected value:** Is the true value(or the label present in your data)

Often machine learning models are not 100% accurate or perfect, they tend to deviate from the true value or expected value.

Explaining with an example: If we are predicting the age of a person based on few input variables or features.

- Our machine learning model predicted the age as 28 years
- However, the actual age of the person is 29 years.
- Here **28 years is predicted value** and **29 years is expected value or true value.** As data scientists, we try to minimize the error while building models.

Cost Function



The difference between the true value and the model's predicted value is called **residual.**

Cost Function Types/ Evaluation Metrics

There are three primary metrics used to evaluate linear models (to find how well a model is performing):

1. Mean Squared Error:
2. Root Mean Squared Error
3. Mean Absolute Error

Mean Squared Error (MSE)

- MSE is simply the average of the squared difference between the true target value and the value predicted by the regression model.

- As it squares the differences, it **penalizes** (gives some penalty or weight for deviating from the objective) **even a small error** which leads to **over-estimation of how bad the model is.**

$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)}^{2}$$

The square of the difference between actual and predicted

DPhi

Root Mean Squared Error (RMSE)

- It is just the square root of the mean square error.

- It is preferred more in some cases because the errors are first squared before averaging which poses a high penalty on large errors. This implies that **RMSE is useful when large errors are undesired.**

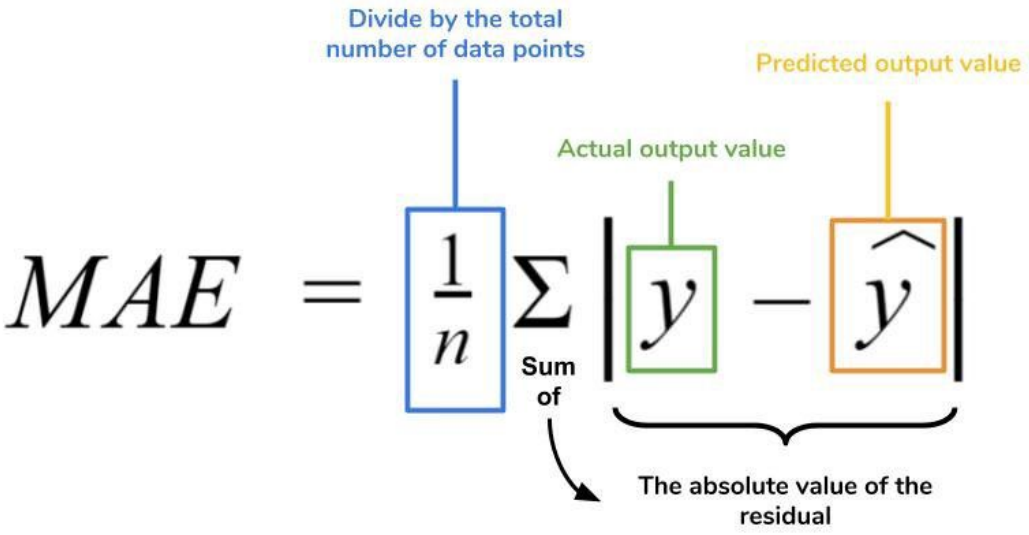
$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

DPhi

Mean Absolute Error(MAE)

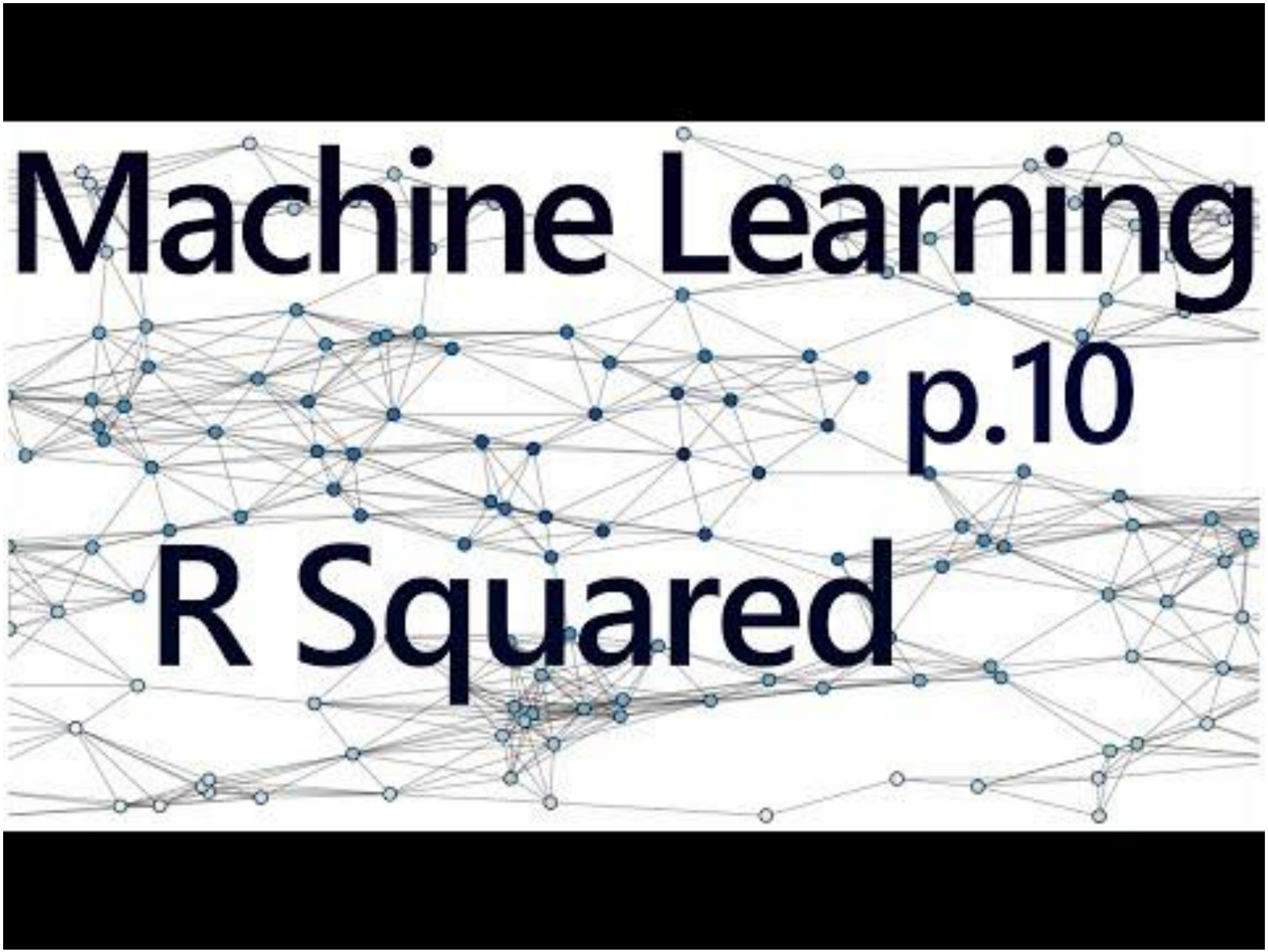
- MAE is the absolute difference between the target value and the value predicted by the model.

- MAE **does not penalize the errors as effectively as mse** making it not suitable for use-cases where you want to pay more attention to the outliers.



DPhi

R Squared (Coefficient of determination)



DPhi

R Squared (Coefficient of determination)

- R-squared is a goodness-of-fit measure for linear regression models.
- It represents the coefficient of **how well the values fit compared to the original values**. The values from 0 to 1 are interpreted as percentages.
- The higher the value is, the better the model is.

$$R^2 = 1 - \frac{\sum(y_i - \hat{y})^2}{\sum(y_i - \bar{y})^2}$$

Where,

$\hat{y}$ – *predicted value of y*

$\bar{y}$ – *mean value of y*

- Going by the name, you might think R2 cannot be negative. However, it can. A Negative R2 means you are doing worse than the mean value.

Which metrics to use when?

Which metrics to use when?

This is an important question and we get used to learning these measures over time. Sharing some resources with you all so that it helps you understand what metrics to be used in the context of solving a regression problem.

- https://medium.com/usf-msds/choosing-the-right-metric-for-machine-learning-models-part-1-a99d7d7414e4
  (you may ignore “Bonus” section in the article for time being)

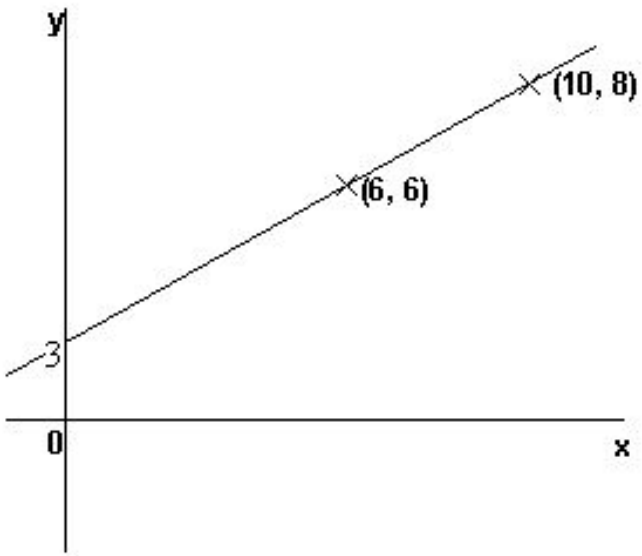
Note: Gradient Descent is a slightly advanced topic. Feel free to skip it if you don’t want to go into maths at all.

Gradient

Gradient is another word for "slope". The higher the gradient of a graph at a point, the steeper the line is at that point. A negative gradient means that the line slopes downwards.

<u>Finding the gradient of a straight-line graph</u>

It is often useful or necessary to find out what the gradient of a graph is. For a straight-line graph, pick two points on the graph. The gradient of the line = (change in y-coordinate)/(change in x-coordinate) .



In this graph, the gradient = (change in y-coordinate)/(change in x-coordinate) = (8-6)/(10-6) = 2/4 = 1/2

We can, of course, use this to find the equation of the line. Since the line crosses the y-axis when y = 3, the equation of this graph is y = ½x + 3 .

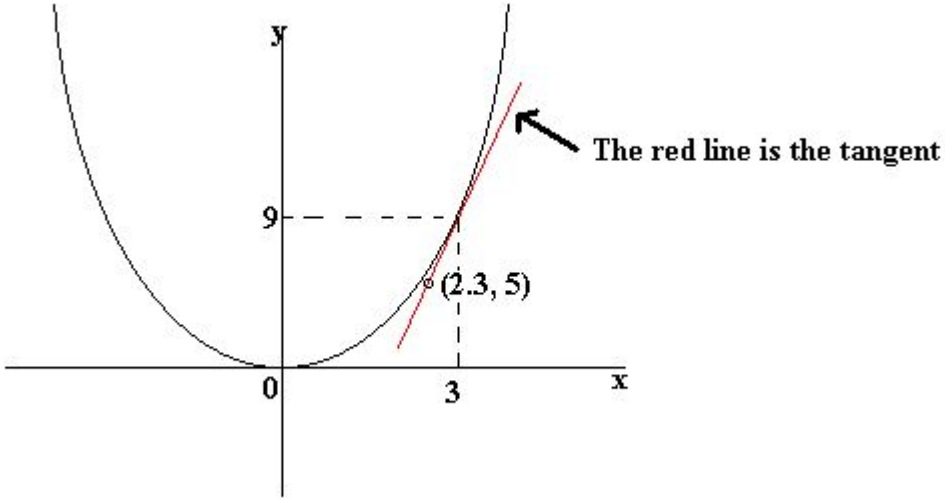
Gradient

Finding the gradient of a curve

To find the gradient of a curve, you must draw an accurate sketch of the curve. At the point where you need to know the gradient, draw a tangent to the curve. A tangent is a straight line which touches the curve at one point only. You then find the gradient of this tangent.

Example

Find the gradient of the curve $y = x^2$ at the point (3, 9).



Gradient of tangent =
(change in y)/(change in x)
= (9 - 5)/ (3 - 2.3)
= 5.71

Gradient Descent

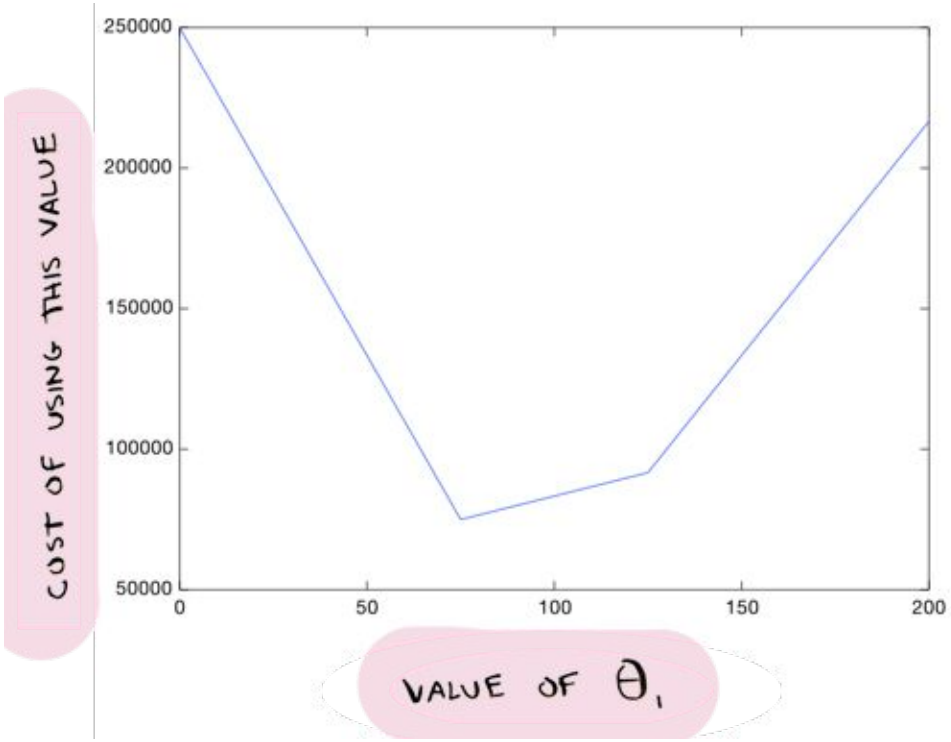
The cost function will tell you how good those values are (i.e. it will tell you how far off your predictions were from the actual data). But what do we do based on that information? How do we find the values of m and b that will draw the best line? By using **gradient descent**.

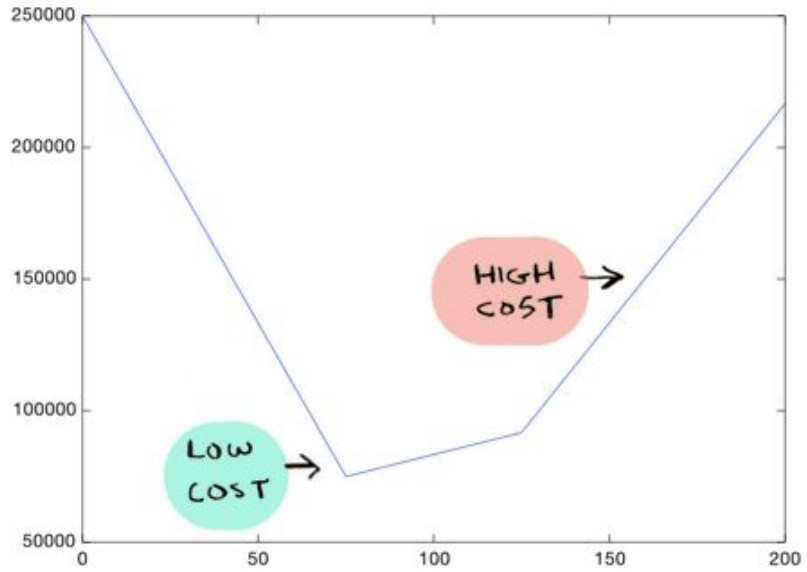
In a nutshell, to update m and b values in order to reduce Cost function (minimizing RMSE value) and achieving the best fit line the model uses Gradient Descent. The idea is to start with random m and b values and then iteratively updating the values, reaching minimum cost.

Let's start with a simpler version of gradient descent, and then move on to the real version.

Gradient Descent

Suppose we decide to leave b at zero. So we experiment with what value m should be, always keeping b at 0. Now you can try various values for m, and you will end up with different costs. You can **plot all of these costs on a graph**:

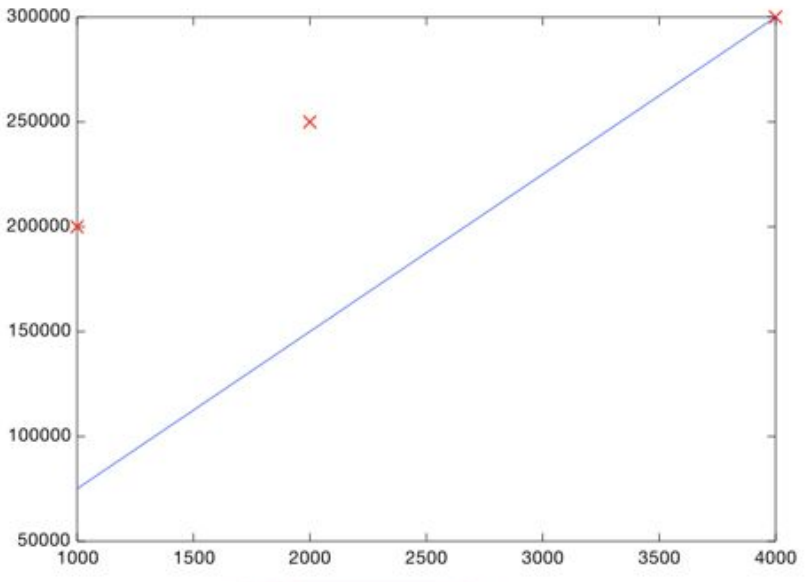




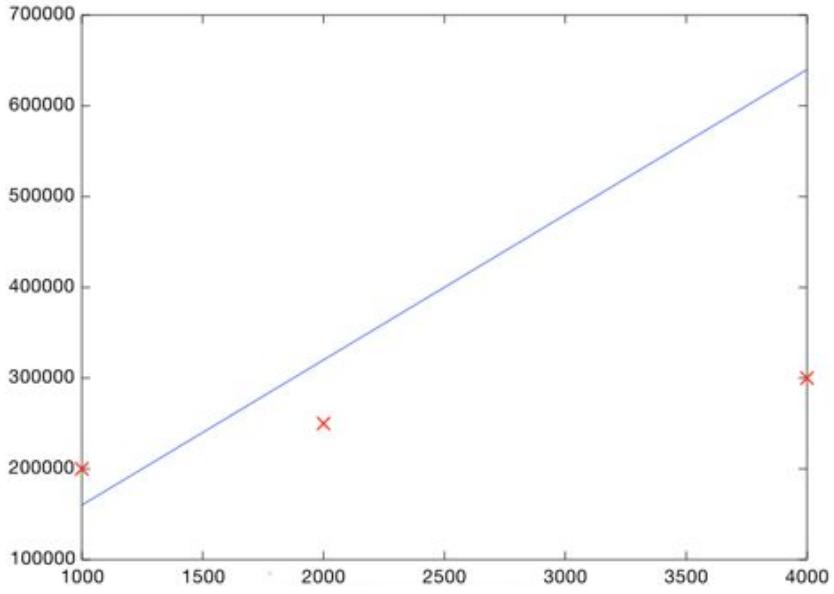
DPhi

Gradient Descent

Here are the corresponding lines (remember, b is zero in these lines):



m = 75

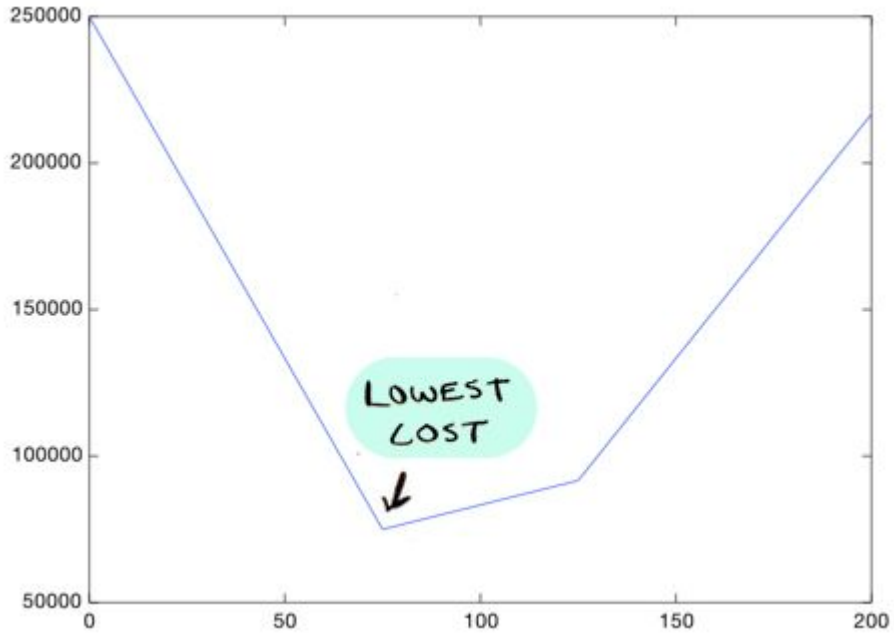


m = 160

We can see that the **line on the left seems to fit the data better than the line on the right, so it makes sense that the cost of that line is lower**. And from this graph it looks like m = 75 gives us the lowest cost overall.

DPhi

Gradient Descent

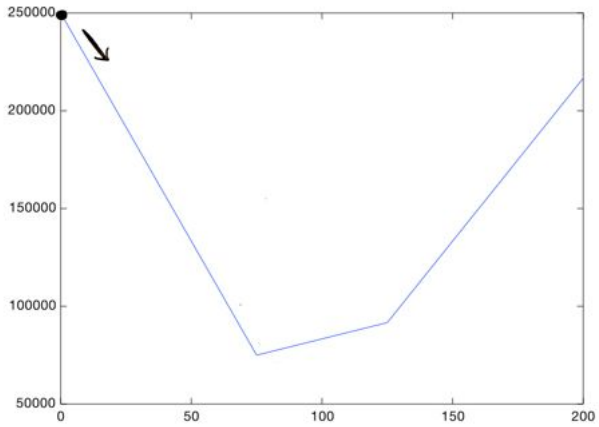


Since it is the lowest point in this graph. So with all the costs graphed out like this, we just need to find the lowest point on the graph, and that will give us the optimal value of m!

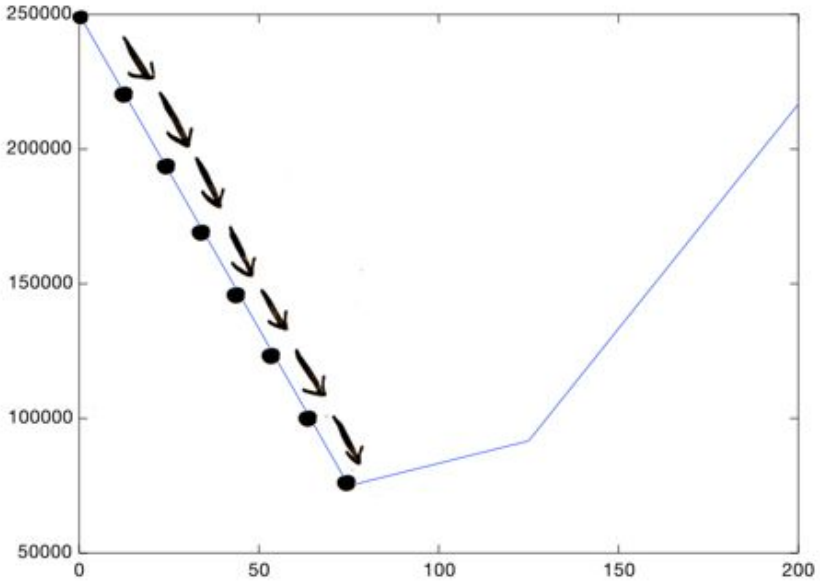
Gradient descent helps us find the lowest point on this graph. You start with a value for m, and update it iteratively till you arrive at the best value. So you can start at m = 0. Then you have to ask, should I go left or right?

Gradient Descent

Well, we want to go down, so lets go right a small step:



This is the new value for m. Again we ask, should we go left or right? At each step, you need to head downward, till you get to a point where you're as low as you can go:



DPhi

Gradient Descent

This is gradient descent: going down bit by bit till you hit the bottom.

How do you figure out which way is down? The answer will be obvious to calculus experts but not so obvious for the rest of us: you take the derivative at that point.

But the important bit to know is, if you take the current value of m and add the derivative at that point, you will go down. You just do that a bunch of times (say 1000 times) and you will hit bottom!

Gradient Descent

The video in the next slide explains the process of gradient descent. You only need to watch till 16:07 to gain some understanding and not go into the python implementation explained later on.

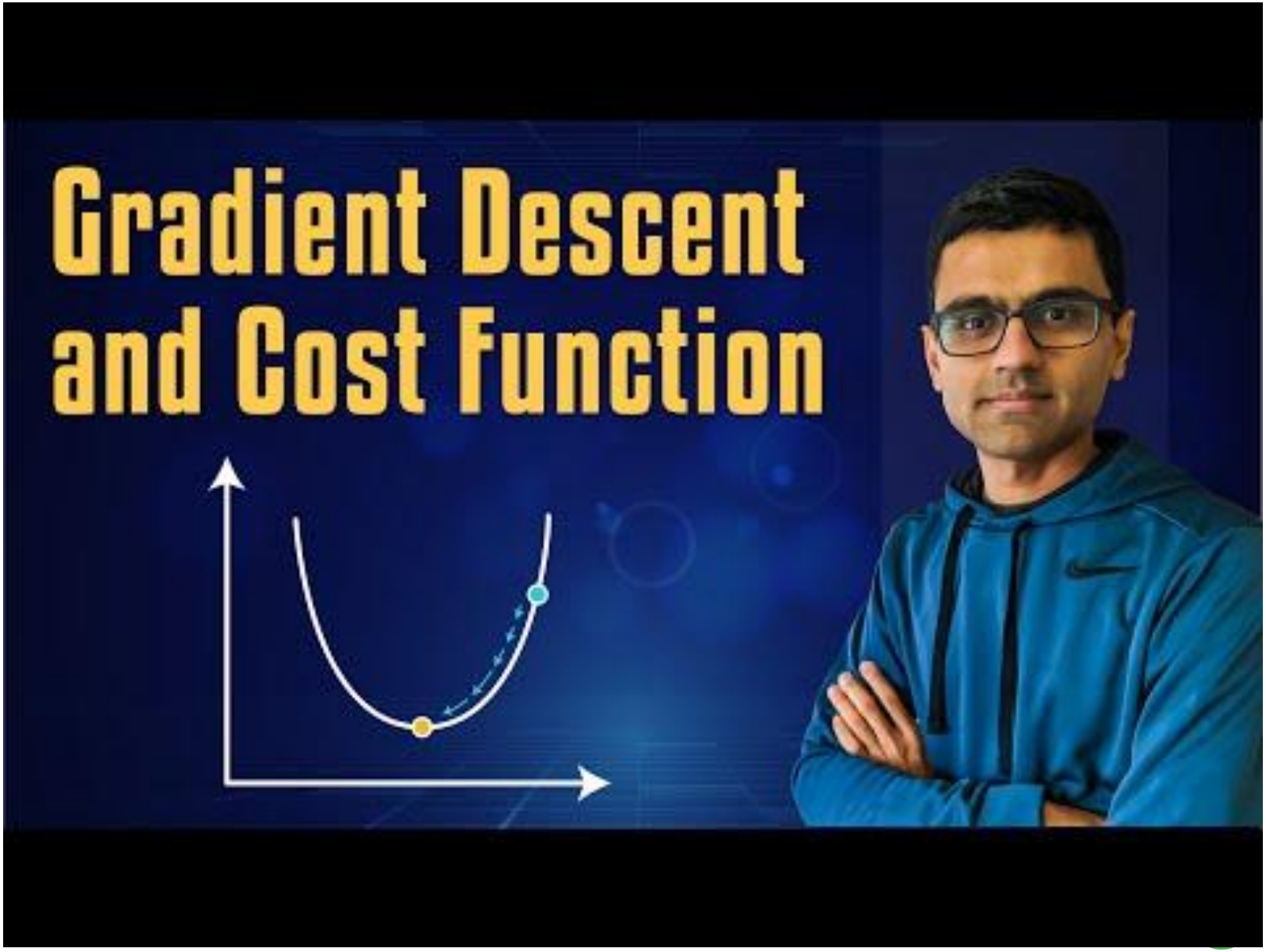
Also, as the instructor said, there's no need to dive into the mathematics or worry about not understanding some math right now.

Gradient Descent

The video in the next slide explains the process of gradient descent. You only need to watch till 16:07 to gain some understanding and not go into the python implementation explained later on.

Also, as the instructor said, there's no need to dive into the mathematics or worry about not understanding some math right now.

Gradient Descent and Cost Function



DPhi

Recap

- ➔ Linear regression is used to predict a value (like the sale price of a house).
- ➔ Given a set of data, first try to fit a line to it.
- ➔ The cost function tells you how good your line is.
- ➔ You can use gradient descent to find the best line.

Slide Download Link

You can download these slides from the below link:

https://docs.google.com/presentation/d/10NzYtmqnoKTGXTvwK51g-zB9zImC3FqBHRdM8oRcpTI/edit?usp=sharing

DPhi

That’s it for this unit. Thank you!

Feel free to post any queries on Discuss.

DPhi