Linear Regression

Machine Learning Mini Course

**What’s all this jargon?**

Some things you might hear in the news or on the street are “artificial intelligence”, “machine learning”, and “deep learning”. Let’s demystify these.

Artificial Intelligence ⊇ Machine Learning ⊇ Deep Learning

**Artificial intelligence** is a broad term referring to any sort of computer program that emulates “intelligence”. Remember Jigga Jigga from the previous years? The competition was to code our AI’s with a strategy so that it would play the game “intelligently”. Remember that we had to code the strategy ourselves, and the AI was just following those instructions.

**Machine learning** is when a machine “learns” the strategy itself, and this is what we will be learning about in this course. You can find a longer explanation below.

**Deep learning** is a type of machine learning that involves a LOT of data and very complex algorithms. You might hear about Google or Facebook doing things with deep learning. This is because it is very powerful, but requires a lot of data that only big companies can more easily get their hands on.

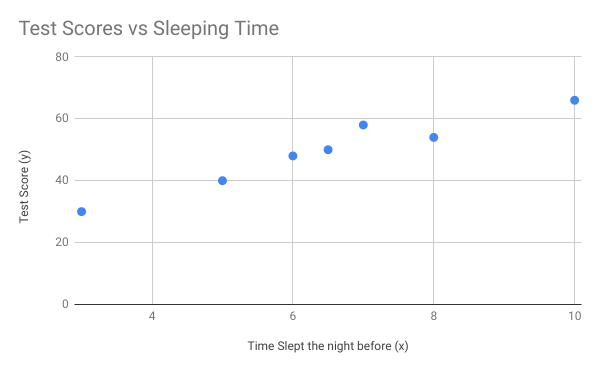
**What is Machine Learning?**

Machine Learning is an approach to solving problems. Humans are very versatile because we can learn from our experiences; we can be thrown in many different situations and figure them out. Conventional computer programs require a programmer to tell them what to do. This can be especially difficult when the programmer is unsure themselves what to do (Example: facial recognition). Machine Learning is then using patterns in the data to figure out an internal model of how things work. Then we use that internal model to predict results on new data.

In this class specifically, we will be looking at a subset of machine learning called **supervised learning**. This means that we have the “answers” to all the data, like how much a house costs or whether an image is a cat or a dog. If you’re curious to know what happens when we don’t have the “answers”, this type of machine learning is called **unsupervised learning**, when we have the machine look for patterns. An example of this is if a store looks at the data from website traffic to try to find trends in buyer activity and maybe tailor the website to their needs. We won’t be going into too much detail on this though. Also, there are still other types of machine learning, like reinforcement learning, GAN’s, and more, but we will just look at supervised learning for now.

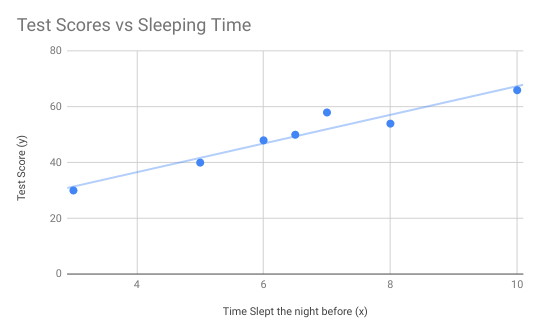
**Example: Linear Regression**

One-variable linear regression takes in an x and tries to output a y. For example, our y could be “score on Chem final” and our x could be “hours of sleep beforehand.”



Then we want to find optimal w, b such that y = wx+b is “close” to the data points. Theoretically, if we could find the best (w, b), we’d be able to predict anyone’s Chemistry scores. Just input their previous’ night sleep, and we’d get a “close” guess.

A good way to visualize this is a “line of best fit.”



**Why Linear Regression?**

Linear Regression might not look that impressive, especially when compared to the other flashy algorithms. Yet linear regression is a viable algorithm within industry; I’d guess that about 30% of industry problems can be solved with linear regression. Furthermore, the Neural Network algorithm is based upon Linear Regression—you can view a neural network as a stack of linear regression models.

**Multivariate Linear Regression:**

Now imagine we had two variables, (x1, x2). Then our model would be y = w1x1 + w2x2 + b. This is a plane in two dimensions. Likewise, we can have as many inputs as we want, and our model would be y = w1x1 + w2x2 + … + wnxn + b

**Loss:**

Now we need a metric to tell how “good” or “close” we’re doing. But instead of seeing how “close” we are, quantifying how “bad” we are is easier. This is called “loss” or “error”. There are many types of loss; we’ll go over mean absolute error and mean squared error.

Mean Absolute Loss =

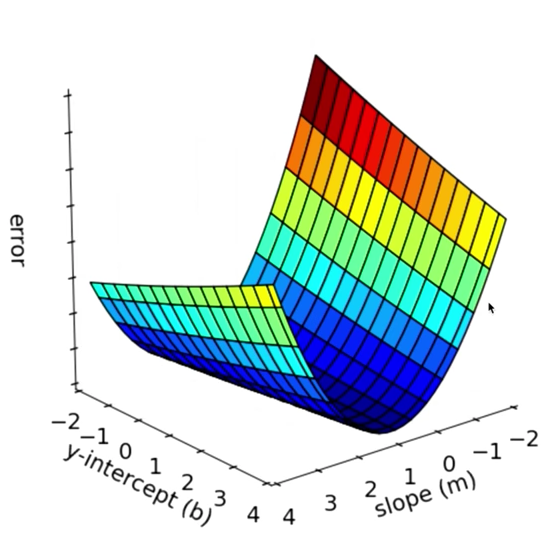
Mean Squared Error =

Where m is the size of the dataset and we sum up over all the true answers and the predicted answers.

**Loss Function:**

But given (w, b), then we can figure out the loss associated with this line. Just take the equation y = wx + b, find the set of predictions ypred, and calculate the total loss. This is what we’ll call the “loss function,” henceforth denoted loss(w, b).

Now, and here’s the hardest part about machine learning, we can define a “loss function space.” With our simple y = wx + b model, this is the 3D space of loss(w, b) plotted against w and b.

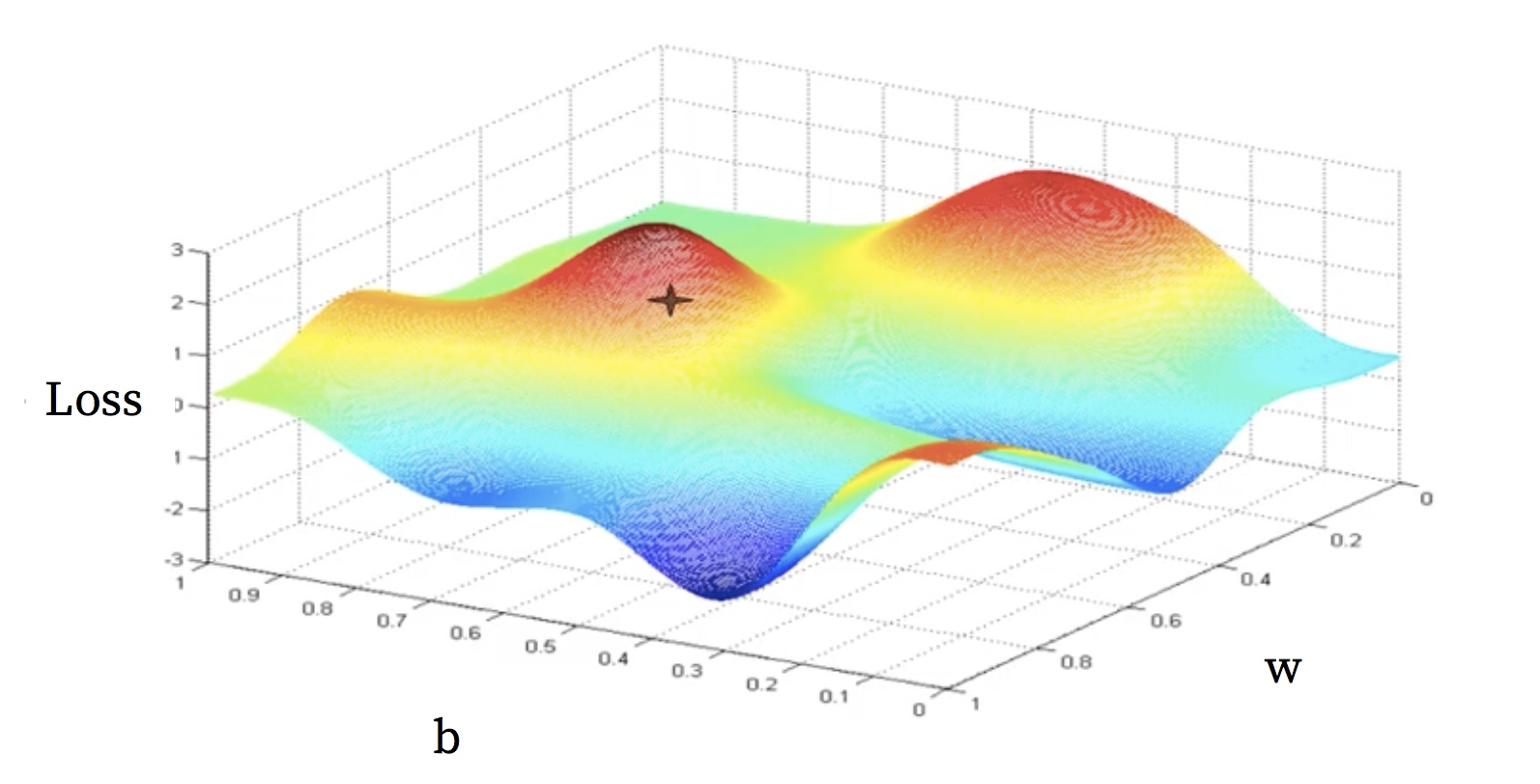


Note: The (m) should be a (w)

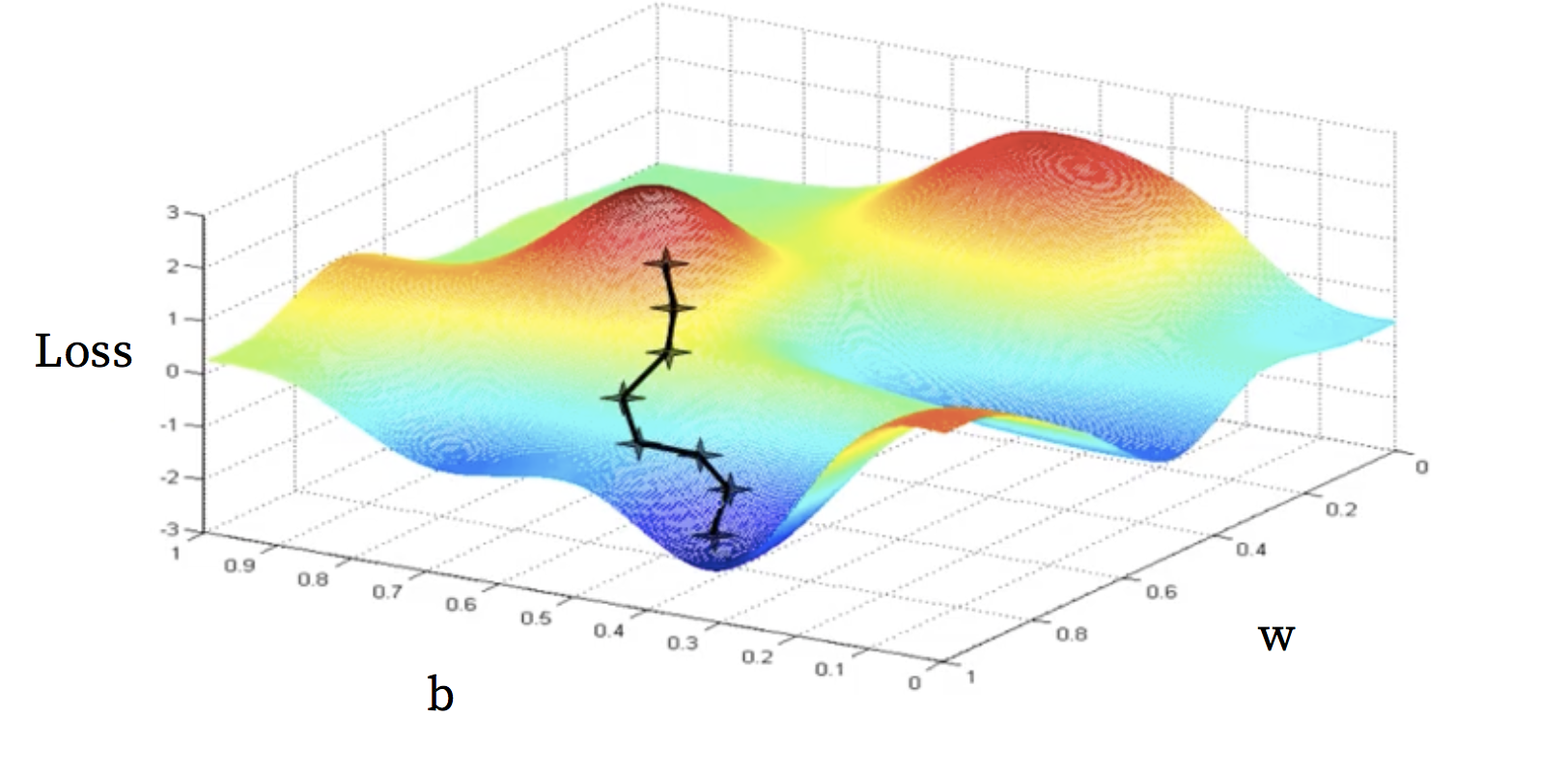
Alright, if things aren’t clear, you can ask either Oliver or Joshua for help; otherwise congratulations; that’s the hardest part of machine learning! We can reframe our original objective as just finding a place where the loss is extremely low.

**Gradient Descent**

Alright, then how can we find a place where the vertical axis is low? Well imagine you’re on a really foggy hill. How can you get to the lowest place possible? [Spooky cliffhanger dun Dun DUN!!!]



Answer: Walk downhill. This ingenious idea is called “gradient descent,” as in descend against the gradient (remember positive gradient means uphill). So if we had the previous loss function space, we’d do this:



Two more things to note. First, each iteration is called a step. There isn’t any continuous “rolling” down the hill, there’s a bunch of small steps. But you can imagine that if you’re step size was too big, you’d miss a good local minimum. Thus, you want your step size to be not too big.

Second, each step in the loss function represents a small improvement of the weights. A good gif explaining this is found here: <https://tinyurl.com/ya5gjfl5>