Machine Learning

Lecture 2

Justin Pearson[1]

[1] https://justinkennethpearson.github.io/teaching/courses/intro_ml/

Today's plan

- A (review) of elementary calculus.
- Gradient Descent for optimisation.
- Linear Regression as Gradient Descent.
- An exact Method of Linear Regression.
- Features and non-linear features.
- Looking at your model.
- Introduction to regularisation.

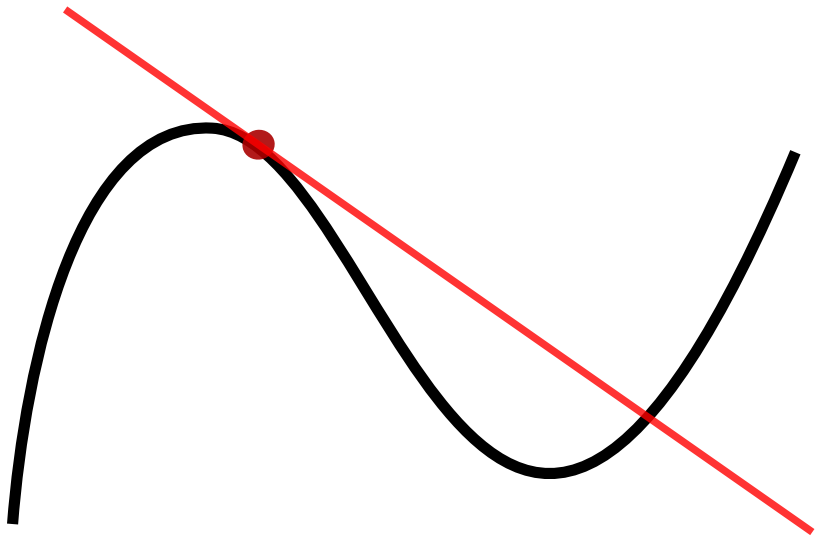
Gradients and Derivatives

Given a continuous function f what does the derivative

$$\frac{\mathrm{d}}{\mathrm{d}x} f(x) = f'(x)\,,$$

tell us?

Tangent Line[2]



The slope of the tangent line is equal to first derivative of the function at that point.

[2] Picture from
https://commons.wikimedia.rg/wiki/File:Tangent_to_a_curve.svg

Gradients Taylor Expansion

For a reasonably well behaved function, f, the Taylor expansion about a point x_0 is the following:

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2!}f''(a)(x - x_0)^2 + \frac{1}{3!}f'''(x_0)(x - x_0)^3 + \cdots.$$

The non-linear terms get smaller and smaller. Thus we could say that around a point x_0

$$f(x) \approx f(x_0) + f'(x_0)(x - x_0)$$

Gradients

What happens when

$$\frac{\mathrm{d}}{\mathrm{d}x} f(x) = 0 \,?$$

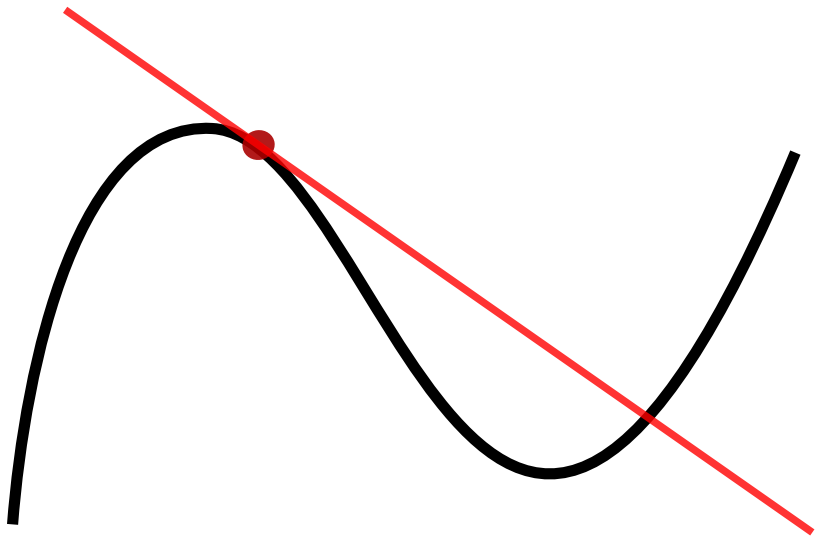
Gradients

What happens when

$$\frac{\mathrm{d}}{\mathrm{d}x} f(x) = 0\,?$$

We are at a minima or an inflection point. To check that it is a true minima we must check if $f''(x) = 0$.

Gradient Descent



If you are at a point, and you go in the direction of the gradient then you should decrease the value of the function.
You are on a hill, you along a vector that has the steepest gradient.

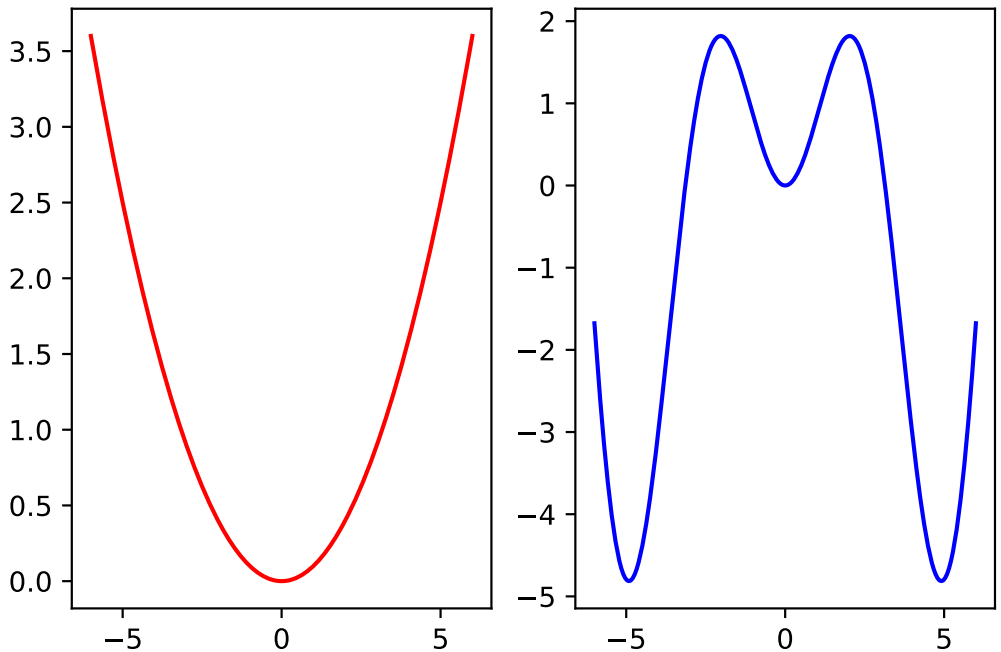
Gradient Descent - One variable

Given a learning rate α and an initial guess x_0

$x \leftarrow x_0$;
while *not converged* **do**
  $x \leftarrow x - \alpha \frac{\mathrm{d}}{\mathrm{d}x} f(x)$;
end

Question, what happens when α is very small and what happens if α is too large?

Minima



The red function on the left only has 1 minimum, while the function on the right as multiple local minima.

Minima

Gradient descent is only guaranteed to find the global minimum if there is only one. If there are many local minima, then you can restart the algorithm with another guess and hope that you converge to a smaller local minima.
Even so, gradient descent is a widely used optimisation method in machine learning

Partial derivatives

How do you differentiate functions of multiple parameters? For example

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

We can compute partial derivatives. The expression

$$\frac{\partial f(x, y)}{\partial x}$$

is the derivative with respect to x where the other variables (y) in this case are treated as constants.
So

$$\frac{\partial f(x, y)}{\partial x} = y + 0 + 2yx$$

Gradient Descent — Multiple Variables

Suppose that we have a function that depends on an n-dimensional vector, $x = (x_1, \ldots, x_n)$ Then the tangent vector or gradient is given by

$$\nabla f(x) = (\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n})$$

Gradient descent works in multiple dimensions, but there is even more of a chance that we can have multiple local minima.

New Notation

Given a data set, x,y of m points we will denote the ith data item as

$$x^{(i)}, y^{(i)}$$

This is an attempt to make expressions like $\left(x^{(i)}\right)^2$ be more understandable. I will try to be consistent.

Linear Hypothesises

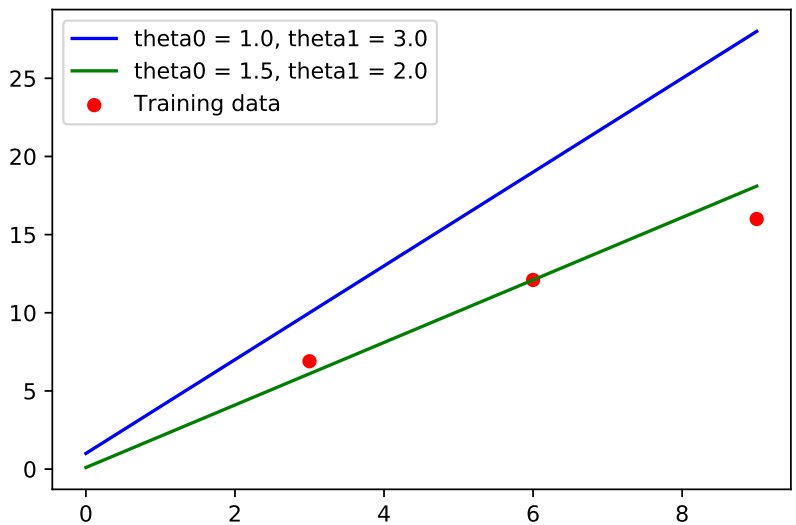
Consider a very simple data set

$$x = (3, 6, 9)$$
$$y = (6.9, 12.1, 16)$$

We want to fit a straight line to the data. Our hypothesises is a function parameterised by θ_0, θ_1

$$h_{\theta_0,\theta_1}(x) = \theta_0 + \theta_1 x$$

Hypothesises



Just looking at the training data we would say that the green line is better. The question is how to we quantify this?

Measuring Error - RMS

Root Mean Squared is a common cost function for regression. In our case given the parameters θ_0, θ_1 the RMS is defined as follows

$$J(\theta_0, \theta_1, x, y) = \frac{1}{2m} \sum_{i=1}^{m} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})^2$$

We assume that we have m data points where $x^{(i)}$ represents the ith data point and $y^{(i)}$ is the ith value we want to predict. Then $h_{\theta_0,\theta_1}(x^{(i)})$ is the model's prediction given θ_0 and θ_1.
For our data set we get

$$J(1.0, 3.0) = 33.54$$
$$J(1.5, 2.0) = 2.43$$

Obviously the second is a better fit to the data.
Question why $(h_\theta(x) - y)^2$ and not $(h_\theta(x) - y)$ or even $|h_\theta(x) - y|$.

Learning

The general form of regression learning algorithm is as follows:

- Given training data $x = (x^{(1)}, \ldots, x^{(i)}, \ldots, x^{(m)})$ and $y = (y^{(1)}, \ldots, y^{(i)}, \ldots, y^{(m)})$
- A set of parameters Θ where each $\theta \in \Theta$ gives rise to a hypothesis function $h_\theta(x)$;
- A loss function $J(\theta, x, y)$ the computes the error or the cost for some hypothesis θ for the given data x,y;
- Find a (the) value θ that minimises J.

Linear Regression

Given m data samples $x = (x^{(1)}, \ldots, x^{(m)})$ and $y = (y^{(1)}, \ldots, y^{(m)})$. We want to find θ_0 and θ_1 such that $J(\theta_0, \theta_1, x, y)$ is minimised. That is we want to minimise

$$J(\theta_0, \theta_1, x, y) = \frac{1}{2m} \sum_{i=1}^{m} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})^2$$

Where $h_{\theta_0,\theta_1} = \theta_0 + \theta_1 x$

Linear Regression — Gradient Descent

To apply gradient descent we have to compute

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

and

$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

Linear Regression — Gradient Descent

For θ_0 we get

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \frac{\partial}{\partial \theta_0} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})^2$$

So how do we compute

$$\frac{\partial}{\partial \theta_0} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})^2 \, ? \tag{1}$$

We could expand out the square term or use the chain rule.

The Chain Rule

$$\frac{\mathrm{d}f(g(x))}{\mathrm{d}x} = f'(g(x))g'(x)$$

If you set $f(x) = x^2$ then you get (since $f'(x) = 2x$)

$$\frac{\mathrm{d}g(x)^2}{\mathrm{d}x} = 2(g(x))g'(x)$$

Linear Regression — Gradient Descent

Using the chain rule

$$\frac{\partial}{\partial \theta_0}(h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})^2 = 2(h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)}) \left(\frac{\partial}{\partial \theta_0}(h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)}) \right)$$

With a bit more algebra and expanding out h

$$\frac{\partial}{\partial \theta_0}(h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)}) = \frac{\partial}{\partial \theta_0}(\theta_0 + \theta_1 x^{(i)} - y^{(i)}) = 1$$

For the partial derivative anything not concerning θ_0 is treated as a constant and hence has a derivative of 0.

So putting it all together we get

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \frac{\partial}{\partial \theta_0} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})^2$$

Which equals

$$\frac{1}{2m} \sum_{i=1}^{m} 2(h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})$$

Linear Regression — Gradient Descent

For θ_1 we go through a similar exercise:

$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \frac{\partial}{\partial \theta_1} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})^2$$

Again we can compute the partial derivative using the chain rule

$$\frac{\partial}{\partial \theta_1} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})^2 = 2(h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)}) \left(\frac{\partial}{\partial \theta_1} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)}) \right)$$

With a bit more algebra

$$\frac{\partial}{\partial \theta_1} (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) = x^{(i)}$$

Linear Regression — Gradient Descent

So our two partial derivatives are:

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)}) = \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})$$

$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)}) x^{(i)} = \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) x^{(i)}$$

Linear Regression — Gradient Descent

Our simultaneous update rule for θ_0 and θ_1 is now

- $\theta_0 \leftarrow \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)})$
- $\theta_1 \leftarrow \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_{\theta_0,\theta_1}(x^{(i)}) - y^{(i)}) x^{(i)}$

Since the error function is quadratic we have only one minima. So with suitable choices of α we should converge to the solution.

Linear Regression — Exact Solution

Remember that at a local or global minimum have that

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = 0 = \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

We can try to solve these two equations for θ_0 and θ_1. In the case of linear regression we can.

Linear Regression – Exact Solution

The details are not important. The reason why you can solve it is much more interesting. When you fix the data. You get two linear equations in θ_0 and θ_1.

$$\frac{1}{m}\sum_{i=1}^{m}(\theta_0 + \theta_1 x^{(i)} - y^{(i)}) = \theta_0 + \frac{1}{m}\sum_{i=1}^{m}(\theta_1 x^{(i)} - y^{(i)}) = 0$$

$$\frac{1}{m}\sum_{i=1}^{m}(\theta_0 + \theta_1 x^{(i)} - y^{(i)})x^{(i)} = \frac{1}{m}\sum_{i=1}^{m}(\theta_0 x^{(i)} + \theta_1 (x^{(i)})^2 - y^{(i)}x^{(i)}) = 0$$

Since you have two equations and two unknowns θ_0 and θ_1 you can use linear algebra find a solution. This generalises to multiple dimensions and is implemented in most numerical packages.

Multiple Dimensions or features

So far we have just had one feature. In general we want to model multiple features $x_1, \ldots, x_n$. Our hypotheses become

$$h_{\theta_0,\theta_1,\ldots,\theta_n}(x_1, \ldots, x_n) = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n$$

We will need vectors. Let $\theta = (\theta_0, \theta_1, \ldots, \theta_n)$ and $x = (1, x_1, \ldots x_n)$. Then our hypotheses is simply the dot produce of the two vectors

$$h_\theta(x) = \theta \cdot x = \sum_{j=0}^{n} \theta_j \cdot x_j$$

Notice that we can factor out the constant by adding an extra feature that is always 1.
The loss or error function is then

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (\theta \cdot x - y)^2$$

Multiple features partial derivatives

The maths is much the same as before (do the derivation to make sure that you understand)

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

Note that you are summing over the m data points $x^{(1)}, \ldots, x^{(m)}$. The $x_i^{(j)}$ is the jth component of the ith data item. Don't forget that by convention $x_0 = 1$.
Again you can have an exact solution with a bit of linear algebra.

Linear Regression — Exact or Gradient Descent

- The exact method involves you doing some linear algebra. With large data sets and a large number of features this might be computationally expensive.
- Gradient descent is often quicker with many features and big data sets.

Also, it illustrates a common theme with machine learning algorithms.

New Features — Scaling

Suppose your data set contains the weight of an person along with the amount of Magnesium in their blood. The body weight in your sample goes from 50kg to 120kg. The common range for the amount Magnesium are from 0.70 to 0.95 nmol/L.
Obviously the body weight measurements dominates your data. In theory your machine learning algorithm will be able to cope, but you can help things along by scaling the data by multiplying the data by a weight. You want all the data to have a similar range and mean.

New Features — Linear Regression

- Given your data set, you are free to make new features. You are trying to fit a linear hypothesis.
- Suppose that your data set had three features x_1, x_2, x_3 you could invent a new feature $x_3 - x_2$. You could process your data and feed the 4 features $x_1, x_2, x_3, x_4 = (x_3 - x_2)$ to the linear regression model.
- Linear combination would not enable you to learn anything new. Why?

But you are not limited to linear features.

Non-Linear features with Linear Regression[3]

With linear regression it is possible fit non-linear polynomials. Suppose that you are trying to fit the polynomial

$$h_{\theta_0,\theta_1,\theta_2}(x) = \theta_0 + \theta_1 x + \theta_2 x^2$$

To your data set.

[3] Don't worry about this too much. It is partly there for entertainment.

Non-Linear features with Linear Regression

One way of thinking is to transform your data into another data set.

$$x \to (x, x^2) \to (1, x, x^2)$$

We add the extra one so we can use the vector $\theta = (\theta_0, \theta_1, \theta_2)$ for the parameters.

Non-Linear features with Linear Regression

Computing the partial depravities

$$\frac{\partial}{\partial \theta_0}(h_\theta(x) - y)^2 \quad \text{and} \quad \frac{\partial}{\partial \theta_1}(h_\theta(x) - y)^2$$

Come out the same as before.

Non-Linear features with Linear Regression

What about

$$\frac{\partial}{\partial \theta_2}(h_\theta(x) - y)^2$$

Again using the chain rule we get

$$\frac{\partial}{\partial \theta_2}(h_\theta(x) - y)^2 = 2(h_\theta(x) - y)\frac{\partial}{\partial \theta_2}(h_\theta(x) - y)$$

Which gives

$$2(h_\theta(x) - y)\frac{\partial}{\partial \theta_2}(\theta_0 + \theta_1 x + \theta_2 x^2 - y)$$

Since everything that is not θ_2 is considered a constant we get that

$$\frac{\partial}{\partial \theta_2}(h_\theta(x) - y)^2 = 2(h_\theta(x) - y)x^2$$

Linear Regression — Look at your model

Explainable AI is very important. Often you do not just a predictor you also want to know why your model makes certain predictions.

- A Linear model is understandable. Looking at features with the largest weights gives you some idea what factors are important in the model.
- Linear models help scientists formulate new theories. If certain features are more important then this helps you to formulate scientific hypotheses.
- Ethical considerations, this is more and more important in AI. Why does my model make certain predictions? Why have I not been granted a bank loan?
- Is my machine learning model sexist or racist? Is my machine model trustworthy. See `https://ec.europa.eu/digital-single-market/en/news/ethics-guidelines-trustworthy-ai`

Regularisation

To avoid over fitting we could add the requirement that the learnt parameters do not get to big.
We can do this by modifying our cost function

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{i=1}^{n} \theta_i^2$$

Again you can do gradient descent or on some cases you can find an exact solution.
We will look at this later in the course.

Linear Regression as Machine Learning Overview

- Gradient descent is a way of minimising continuous functions. It works best if the is only one minimum, otherwise you could get stuck in a local minimum.
- Linear regression find the best straight line through a number of data points.
- By looking at the error or loss function you can set up linear regression as a minimisation problem that you can solve with gradient descent. You can prove (although we don't) that there is only one minimum.

Important: Gradient descent is not a good way to solve linear regression problems. It is just to illustrate a common technique in machine learning.

Machine Learning Concepts

- Loss/Error functions measure how well your model fits the training set.
- The loss/error function depends on the parameters of the model. For example the weights of a neural network or the coefficients of your linear function.
- One technique to minimise an error function is to use gradient descent. You use partial derivatives to work out the gradients.
- Regularisation (more on this) is one way of avoiding over fitting.