## **Generalized Regression**

In linear regression, you had encountered problems where the target variable y was linearly related to the predictor variable X. But what if the relationship is not linear? Let's see how we can use **generalized regression**to tackle such problems.

While practicing advanced regression, a key skill is to be able to identify the nature of a function by visualizing the data. For a quick refresher on plots of common functions, [you can visit this page.](https://www.mathsisfun.com/sets/functions-common.html)

You may recall that the general expression for a polynomial function is: f(x)=a0+a1x+a2x2+a3x3+...anxn.

If n=2, it is called a quadratic or a second-degree polynomial; if n=3, it is called a cubic or a third-degree polynomial.

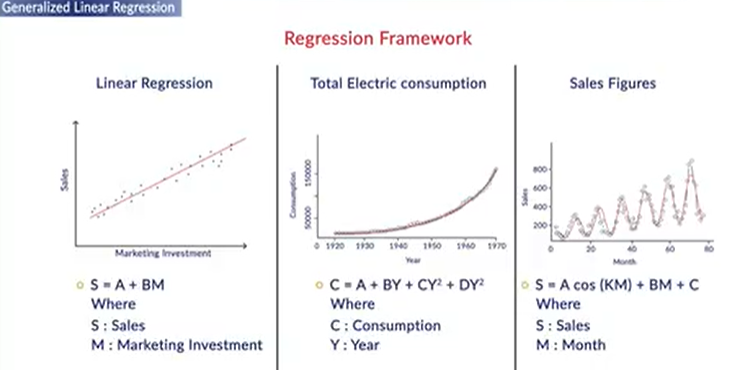
Also, recall that the **roots of a polynomial** f(x) represent the values of x at which the function cuts the x-axis and also remember that a polynomial function can have both **real or imaginary roots.**The real roots (also termed as zero’s) are the once for these values the polynomial equation function will be zero i.e for which ‘x’ f(x) = 0. In other words, where the function f(x) intersects the x-axis in a graph are the real roots. Check khan academy video [here](https://www.khanacademy.org/math/algebra-home/alg-polynomials/alg-finding-zeros-of-polynomials/v/finding-roots-or-zeros-of-polynomial-1).

For example, the quadratic function f(x)=x2−5x+6 has two real roots: x=2,3, though the function f(x)=x2+2x+10 does not have any real roots (and does not cut the x-axis).

**The generic process consists of the following two steps**:

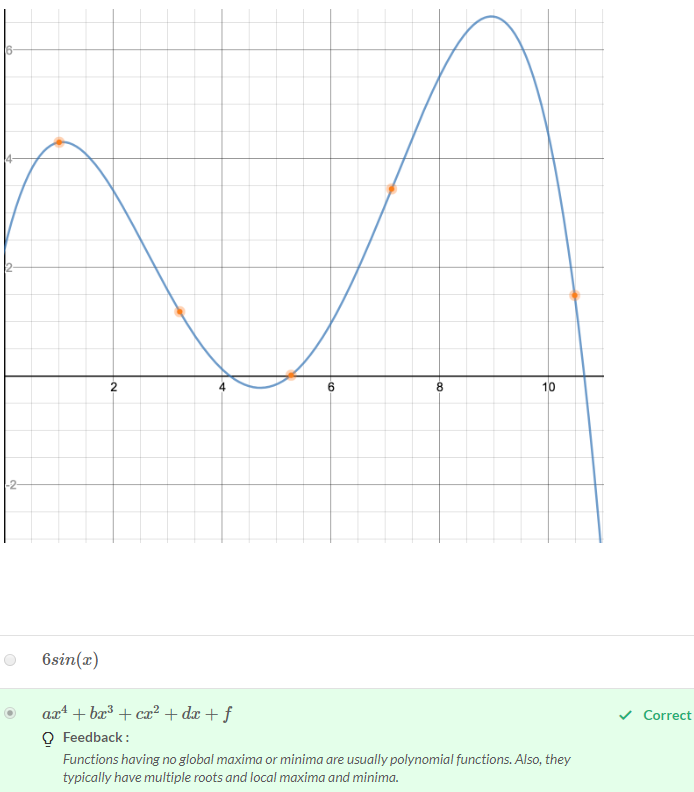
1. Conduct exploratory data analysis by examining scatter plots of explanatory and dependent variables.
2. Choose an appropriate set of functions that seem to fit the plot well; build models using them; and compare the results.

|  |  |
| --- | --- |
|  | Where M is the month, A and B are the variables |



**Profit Function**

Which of the following equations best represents the following graph?



While constructing the regression model, instead of using the raw explanatory variables in the current form, we create some function of the explanatory variables to best explain the data points.

**Feature Generation**

Can x1.x2.x3 be a feature if the raw attributes are x1, x2, x3 and x4?

Top of Form

**Yes**

**Feedback :**

*Derived features can be created using any combination of the raw attributes (linear or non-linear). In this case, the combination x1. x2. x3 is non-linear.*

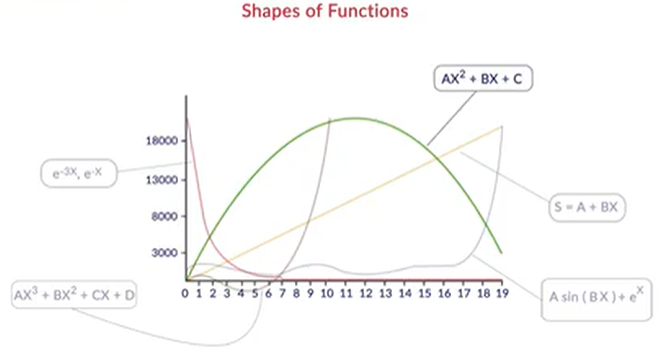
**Feature Generation**

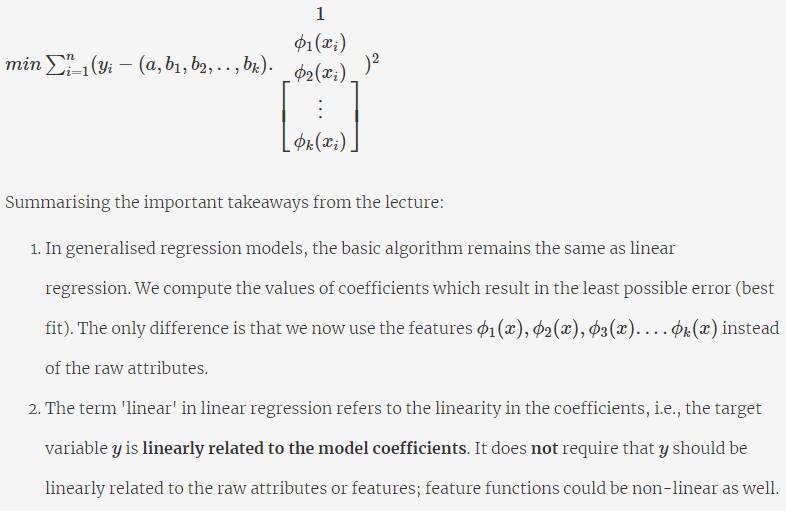
How many maximum features can be created if we have d raw attributes for n data points? Note that (nr) here refers to the number of ways of selecting r items from a set of n.

**Infinite**

**Feedback :**

*You can (in principle) create as many features as you want.*

Bottom of Form



**'Linear' in Linear Regression**

The second point mentioned above is so important (and so often confused) that it is worth elaborating: The model is called 'linear' because the targety**is linearly related to the** **coefficients**.

To fully understand this, it is crucial to note that in regression, the**coefficients** a0,a1,a2,...,ak are your **variables**, i.e., you are trying to find the optimal coefficients that minimize some loss function. On the other hand, the **features** ϕ1(x),ϕ2(x),ϕ3(x)....ϕk(x) are actually **constants** because you are already given the dataset (i.e., the values of x, and hence ϕ(x), are fixed; so, what you are trying to tune are the coefficients).

Thus, saying that 'y is linearly related to the coefficients' implies that **only two operations**can be applied between the coefficients: 1) **Multiplying them by constants** (i.e., the features) such as a1ϕ1(x),a2ϕ2(x) and 2)**Adding the terms**with each other such as a1ϕ1(x)+a2ϕ2(x). What you **cannot** do is multiply them together, raise to one another's power, etc. That is, you cannot have terms such as a0.a1,aa32 etc.

# Systems of Linear Equations

Suppose you have a dataset with **three raw attributes:** weight (x1), age (x2), and height (x3) and a target variable blood pressure (y). Each data point can be expressed as a tuple (X,y) where X=(x1,x2,x3). For simplicity, let's assume that you have only **five data points** in your dataset denoted as (X1,y1)−(X5,y5)**.**Further, you decide to use three feature functions in your model as follows: ϕ1(X)=weight/height, ϕ2(X)=age, ϕ3(X)=√age .

The **training dataset** (after using feature functions) looks like this:

| **Datapoint** | ϕ1(X) | ϕ2(X) | ϕ3(X) | y |
| --- | --- | --- | --- | --- |
| X1 | 22.5 | 34 | 5.8 | 130 |
| X2 | 20.0 | 23 | 4.8 | 120 |
| X3 | 19 | 45 | 6.7 | 125 |
| X4 | 23.5 | 19 | 4.4 | 128 |
| X5 | 22.0 | 21 | 4.6 | 140 |

The linear regression model is defined as y=a0+a1ϕ1(X)+a2ϕ2(X)+a3ϕ3(X). This implies that the model should (ideally) satisfy this relationship for each of the five data points:

a0+a1ϕ1(X1)+a2ϕ2(X1)+a3ϕ3(X1)=y1

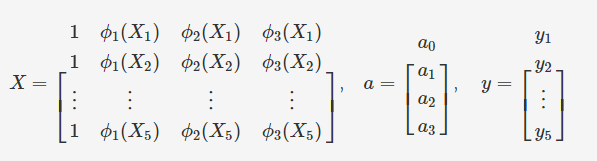
a0+a1ϕ1(X2)+a2ϕ2(X2)+a3ϕ3(X2)=y2

⋮                  ⋮                         ⋮                         ⋮                     ⋮

a0+a1ϕ1(X5)+a2ϕ2(X5)+a3ϕ3(X5)=y5

 You already know the values of all the ϕi(Xj) terms, for e.g., ϕ1(X1)=22.5,ϕ3(X4)=4.4 etc. This is a **system of linear equations** with the variables a0,a1,...,a5.

Solving this system means to find the set of parameters a0,a1,...,a5 which satisfies all the equations. This can be done efficiently using **matrices**(and is done that way by many libraries)**.** The alternate way is to use optimization methods such as gradient descent.

You can write this system into a compact matrix form as follows: Xa=y , where

Thus, solving the system of linear equations boils down to solving the matrix equation Xa=y, i.e., to find a vector a which satisfies Xa=y. This solution can be computed as a=X−1y, where X−1 is the inverse of the matrix X  (but read below).

However, there is a severe problem: X is a **non-square matrix**(and it almost always is; you'll always have more data points than features, i.e., more rows than columns). The problem is that non-square matrices are **non-invertible,**i.e. X−1 simply does not exist**.**Even if X were a square matrix, it is not necessary that X−1 exists: Such matrices are called [**singular matrices.**](https://en.wikipedia.org/wiki/Invertible_matrix)You will study these concepts in the next session; for now, it is sufficient to understand that X−1 does not exist, so we need to find an alternate way to solve the system Xa=y.

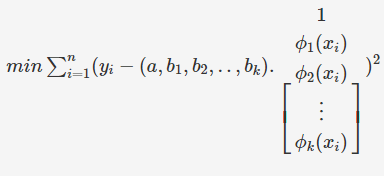
Intuitively, this problem simply reflects a fundamental flaw in our assumption: We have assumed that a vector a exists which satisfies all the five equations **exactly**, but that is rarely the case. In most cases, there will be no such vector a that satisfies the equations exactly for all the data points in the training set (if there is, it corresponds to a very special case of total cost=0). Thus, to solve this problem practically means to find an **approximate solution**to the system Xa=y. The details are beyond the scope of this course, but here's the solution: It turns out that the closest approximation of X−1 (for non-square matrices) is (XTX)−1XT. Thus, the (approximate) solution to this system is given by:

                                                                   a=(XTX)−1XTy

If you want to study this in detail, [you can go through this Khan Academy video.](http://www.youtube.com/watch?v=MC7l96tW8V8&list=PL39469144F25ACECE&index=3) This technique is called the **least squares approximation.**If you want to understand this concept from a more fundamental linear algebra point of view, you should [go through this video by 3Blue1Brown.](https://www.youtube.com/watch?v=uQhTuRlWMxw&t=1s)

**Additional Resources**

* [Khan Academy: Showing that XTX is an invertible matrix](https://www.khanacademy.org/math/linear-algebra/matrix-transformations/matrix-transpose/v/lin-alg-showing-that-a-transpose-x-a-is-invertible)

To summarise, we saw that the objective is to find the optimal model parameters a,b1,b2,...,bk which **minimise the total loss**: 

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  |  |