Contents

Linear and Logistic Regression

Let us refresh our memory about two important classes of data science problems; Regression is where you explain an unknown numeric variable as a function of several known variables. In classification, an unknown categorical variable is computed as a function of known variables. So, the goal in classification is binning the data.

Typically, the unknown variable is the most difficult for the business to compute but has a lot of value (like how much revenue the customer is likely to give in the next 12 months, which of the customers are likely to buy etc.). The known variables on the other hand can be easily measured (like age, gender of the customers).

The business measures both known and unknown variables for some time and then as a data scientist you build models that help them guess accurately the unknown variables.

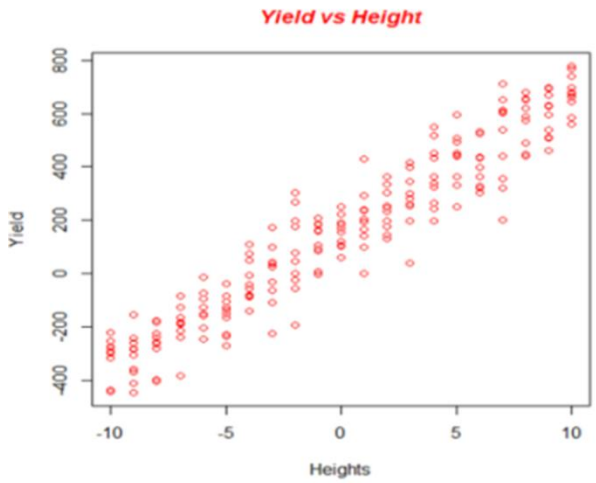
We will in this class study two of the most popular data science models namely linear regression and logistic regression. Linear regression as the name suggests is a regression model. Logistic regression, however, is a classification problem. It is just named that way.

Let us learn three different aspects of both these techniques.

- In scientific aspects, let us gain mathematical intuition of these techniques. As an engineer, you may never code these models yourself. But, the intuition is very helpful.
- In engineering aspects, let us get hands-on and build some models on real data sets.
- In consulting aspects, let us learn to interpret the outcome in a systematic manner.

Linear regression

An agricultural scientist is measuring how the profit of a specific seed varies as a function of ground height from the sea level.

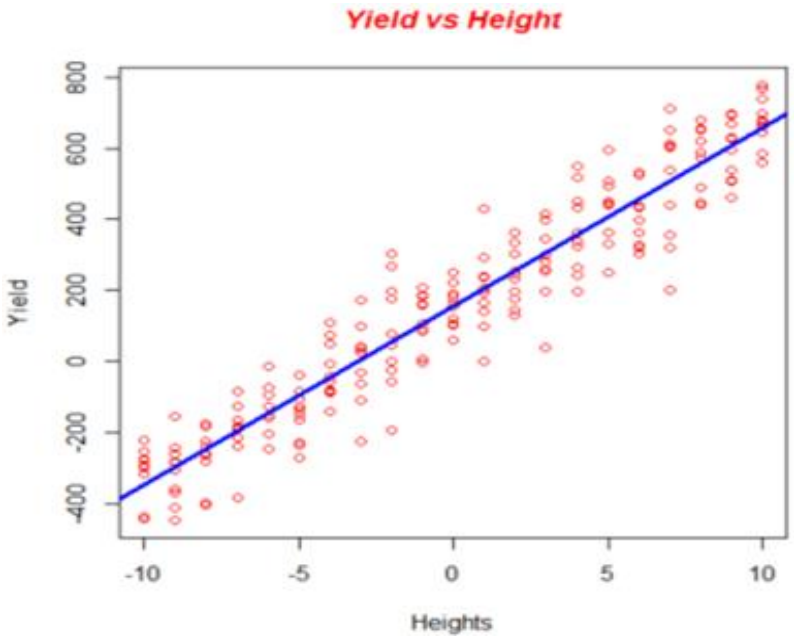


Yield (Profits/acre) as a function of ground level from sea (meters)

Now, as a data scientist, you need to build a model to compute the difficult to measure, unknown variable yield as a function of easy to measure known variable, ground level.

We are working with only one known variable in this case. It is called univariate regression. While in most real life situations you will have more than one known variable (in this case, the rain fall, type of sand etc. will also most likely influence the outcome), the univariate case is easy to plot and understand. So, let us learn this and then move to multivariate cases.

The goal is to plot a line that fits best through this data.



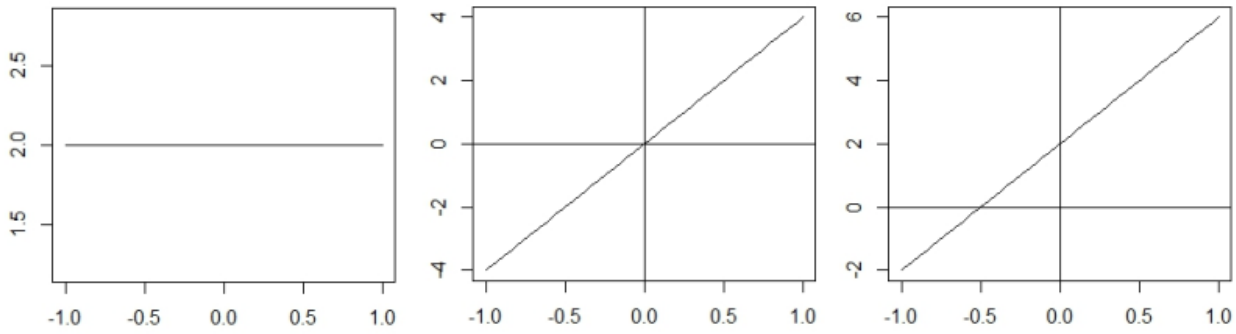
Linear model through the data

How does one fit this line? As our goal is to predict the yield as a function of height, we want to fit a line that gives the closest possible yields given heights.

A possible equation is

$$y = \theta_0 + \theta_1 x$$

For each possible value of θ_0 and θ_1, we get a different line as shown below



$$Left:\ \theta_1 = 0 \quad Center:\ \theta_0 = 0\ and\ Right: Both\ are\ non\ zero$$

The goal is then to pick those two parameters such that for every data point, we make the best possible prediction.

In machine, learning each possible combination of θ_0 and θ_1 is called a hypothesis. So, there is an infinite space of possible hypotheses. We use a learning algorithm like linear regression to identify the best hypothesis.

The learning algorithm in this case should help us choose the right set of θ_0 and $\theta_1$that minimizes the discrepancy between the predicted and actual values. Let us define a function called cost function which is the difference between the predicted and original values

Cost function,

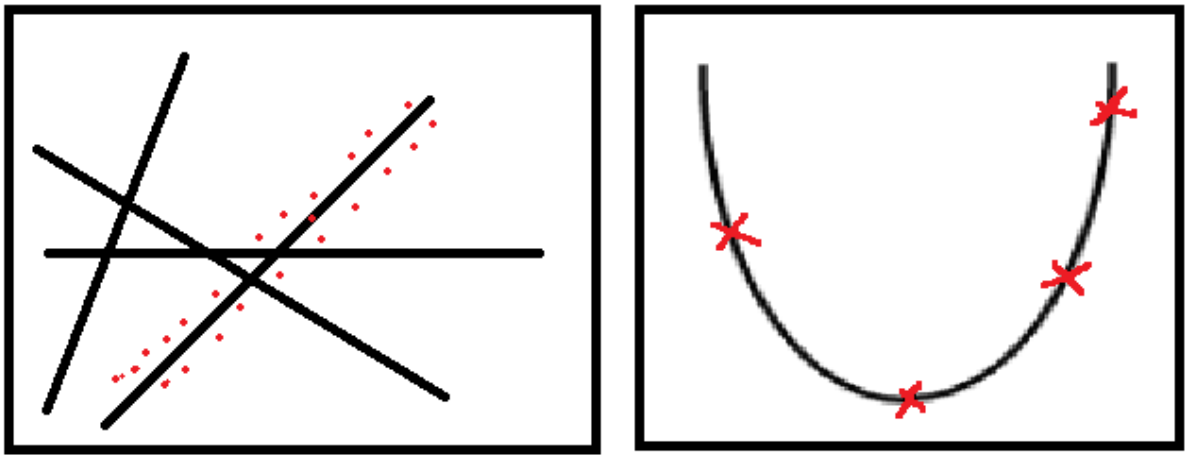
$$J = \frac{1}{2m}\sum_{1}^{m}((\theta_0 + \theta_1 x_i) - y_i)^2$$

We square the term to ensure that positives and negatives do not get cancelled. So, the cost function is the average of the squares of differences of predicted and real values of all points. 2 in the denominator makes the mathematics easier.

Let us get more intuition with a single parameter hypothesis and cost functions.

$$h = \theta_1 x$$

$$J = \frac{1}{2m}\sum_{1}^{m}((\theta_1 x_i) - y_i)^2$$

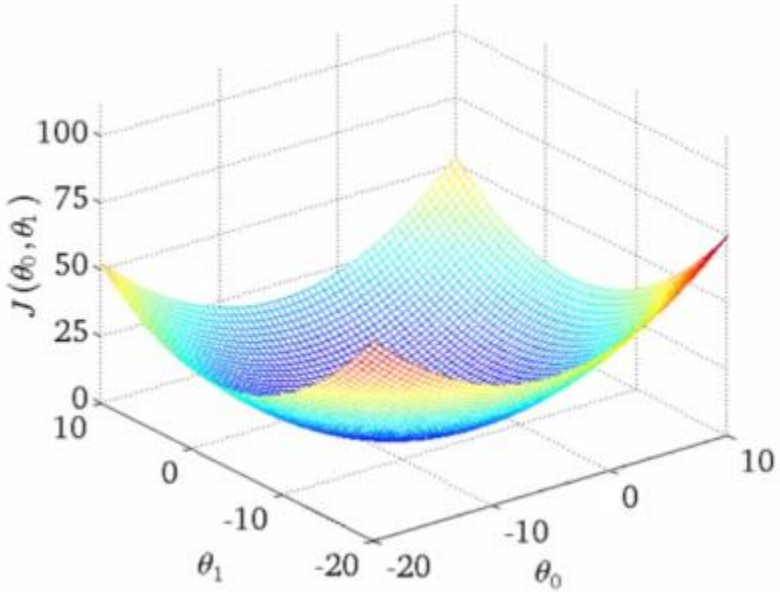


Various hypothesis functions for various parameters

Associated cost function

Note that while hypothesis function is dependent on attributes, cost function is dependent on parameters. Hypothesis function is linear and cost function is quadratic.

For two parameters, the cost function will be something like

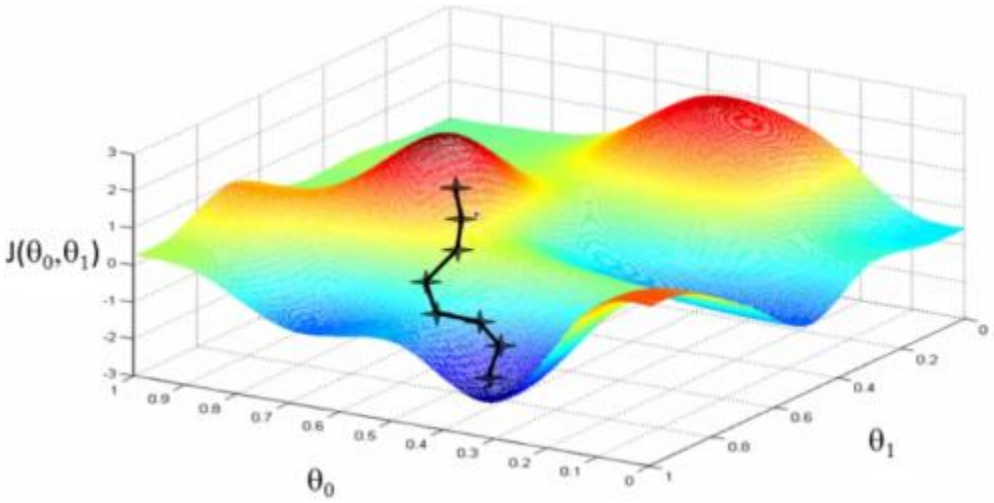


The algorithm should choose the parameters to minimize the cost function.

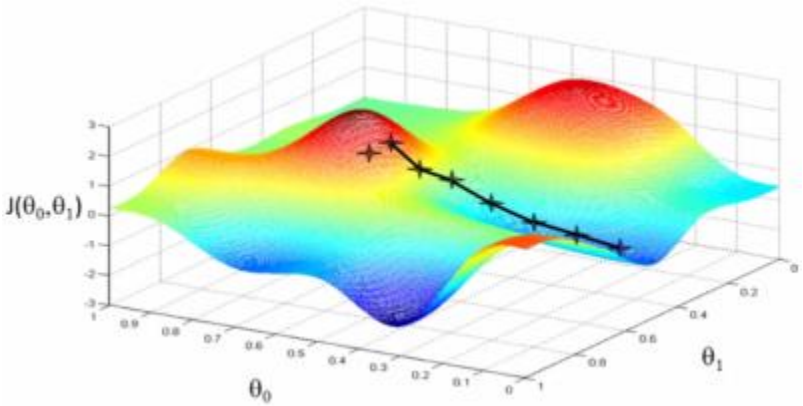
For the linear regression, there is an analytical way to minimize the cost function. Where you take the partial derivative of cost

functions with respect to θ_0 and θ_1 and solve them simultaneously.

However, defining a cost function and minimizing it is a much more general problem in machine learning and for most of the cost functions, there are no analytical solutions. A very standard algorithm used for finding minimum in such cases is gradient descent. Gradient means slope. Hence, gradient descent is moving in the direction of maximum slope.



But, depending on where we start, we can take different paths.



Mathematically, what we are repeating until convergence

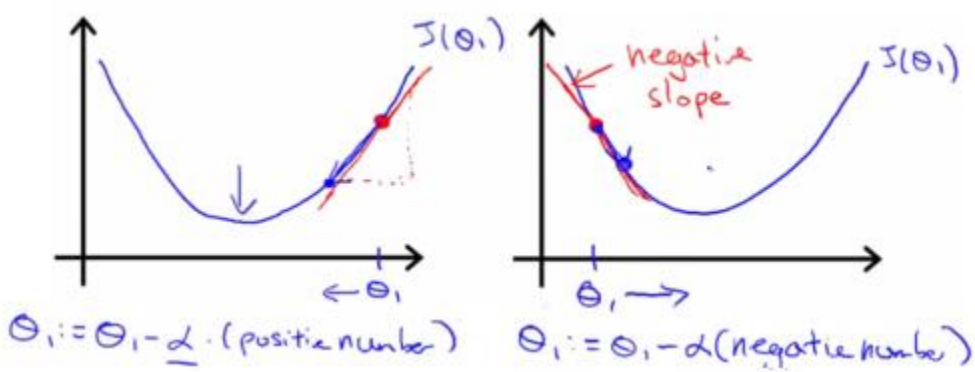
$$\theta_j^{n+1} = \theta_j^n - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0^n, \theta_1^n) for\ j = 0\ and\ 1$$

We update θ_0 and θ_1 simultaneously. $\alpha$is called the learning rate. It tells how big a step we must take at each point. It is called the learning rate. Those of you familiar with calculus can verify that for linear regression, this becomes

$$\text{repeat until convergence } \{$$
$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)$$
$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) \cdot x^{(i)}$$
$$\}$$

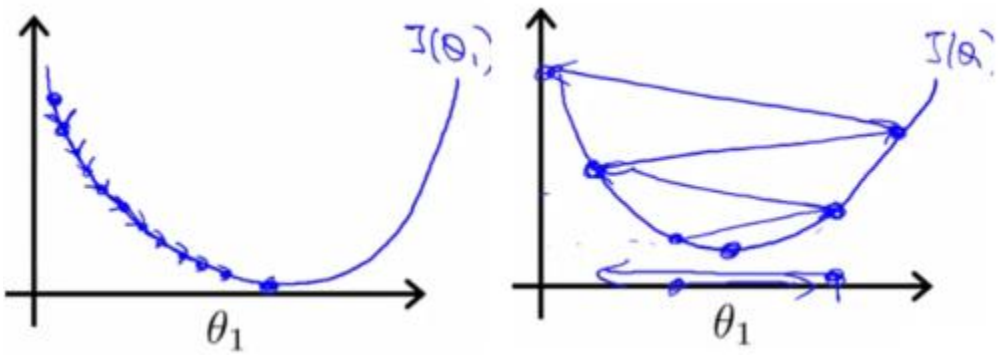
For the linear regression, the cost function will be bow shaped and hence we will always reach the same minimum if we use gradient descent. But, in most general cases, gradient descent will find only a local minimum.

Let us understand gradient descent a bit more again with a single parameter example.



Clearly, gradient descent always moves you towards minima. At minima, as derivative is zero, it keeps you unmoved.

A low learning rate means baby steps. A high learning rate means non-convergence as shown below. However, once you finalize on a learning rate, you need not change it as the slope reduces as you go near the minimum value.



This is in fact a disadvantage of gradient descent that you need to experiment and find the best learning rate. But, if you normalize data and choose a good learning rate through experimentation, you converge fairly well.

There are more advanced and complex algorithms for finding the minima of a cost function. But, gradient descent is the most simplest and popular.

One advantage is that if you have multiple features, gradient descent can be easily scaled up. Let us say, in our original problem, we added rain fall, type of soil also in our analysis. The hypothesis now takes the following form

$$h = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

To simplify this, let us use the matrix notation. Let us introduce a term $x_0$which is always equal to 1. Then, we introduce two column vectors X and θ.

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} \quad x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \theta^T = [\theta_0 \quad \theta_1 \quad \theta_2 \quad \theta_3]$$

So, the hypothesis for a multivariate regression can be written as $\theta^T X$. The gradient descent equation will then be

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)}$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)}$$

$$\dots$$

Actually, for linear regression, the cost function can be minimized analytically by taking derivatives with respect to each of the parameters and simultaneously solving them. Let us say, we are working on the following data

x_0	x_1	x_2	x_3	x_4
1	2104	5	1	45
1	1416	3	2	40
1	1534	3	2	30
1	852	2	1	36

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}$$

Using linear algebra, we can show that the analytical solution is

$$\theta = (X^T X)^{-1} X^T y$$

However, gradient descent works best for very large dimensional problem (attributes more than 10,000). It also is used extensively for other cost functions (logistic regression, neural networks etc.).

When there are categorical variable, they are directly implemented if they are binary. For variables with more bins, dummy variables are created.

Great! That is all the science you need to know about linear regression. Let us do some R. Over to code!

Analyzing linear regression.

In the car.test.frame data, let us plot Disp. as a function of HP and Weight

```
> summary(lm(Disp.~HP+Weight, data=cars))

Call:
lm(formula = Disp. ~ HP + Weight, data = cars)

Residuals:
    Min      1Q  Median      3Q     Max
-57.260 -19.069  -1.341  13.365  92.406

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -88.78832   21.63778  -4.103 0.000131 ***
HP            0.85894    0.18047   4.760 1.37e-05 ***
Weight        0.04680    0.01128   4.150 0.000112 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 27.76 on 57 degrees of freedom
Multiple R-squared:  0.7461, Adjusted R-squared:  0.7372
F-statistic: 83.77 on 2 and 57 DF,  p-value: < 2.2e-16
```

Firstly, the R square is a good indication of how good the fit is. It says here that, we explain 73.7% of the variance in the data. So, the model reduces the variance in the data by 73.7% compared to the simple mean.

The equation is

$$\text{Disp.} = -88.79 + 0.85*\text{HP} + 0.04*\text{Weight}$$

So, both HP and Weight have a proportional relation.

If one of them is negative, that meant it had an inverse relation.

As the coefficient of HP is larger than that of Weight, change in HP has a higher impact that change in Weight on the displacement. When you have multiple variables, analyzing coefficients is a good way to see which attribute impacts the dependent attribute most. For such analyses, it is better to normalize all the dependent attributes.

The standard error is the SD of each of the predicted values. The Pr indicates how confident the model is about the sign of the coefficients and intercept. So, low Pr indicates good and

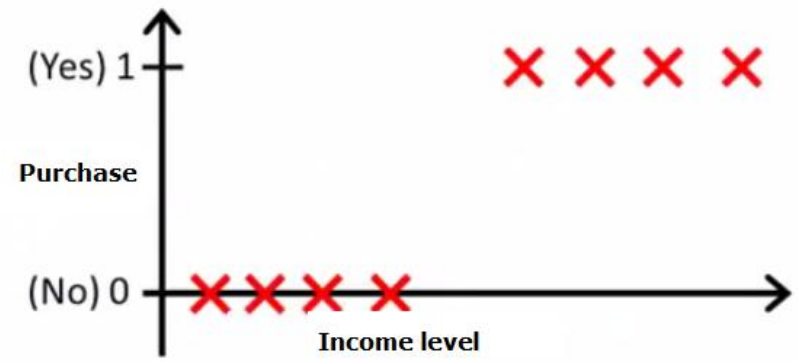
confident fit. T value indicates how many SDs away the value is from zero. A high t value is better.

Logistic regression

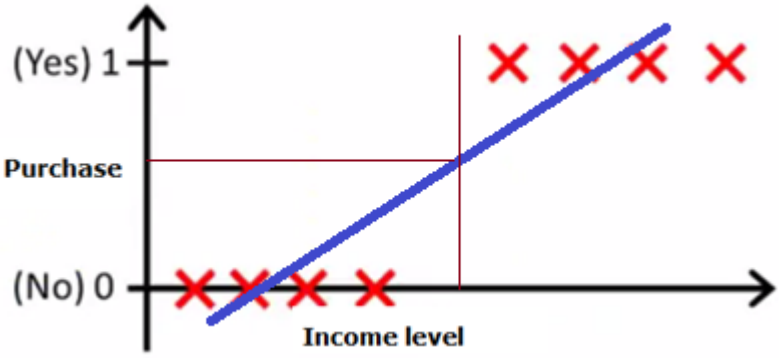
As mentioned before, logistic regression is a classification problem.

Logistic regression extends the ideas of linear regression to the situation where the dependent variable Y, is categorical. We can think of categorical variable as dividing the observations into classes.

For example, if Y denotes whether a particular customer is likely to purchases a product (1) or not likely to purchase (0) we have a categorical variable with 2 *categories or classes (0 and 1)*.

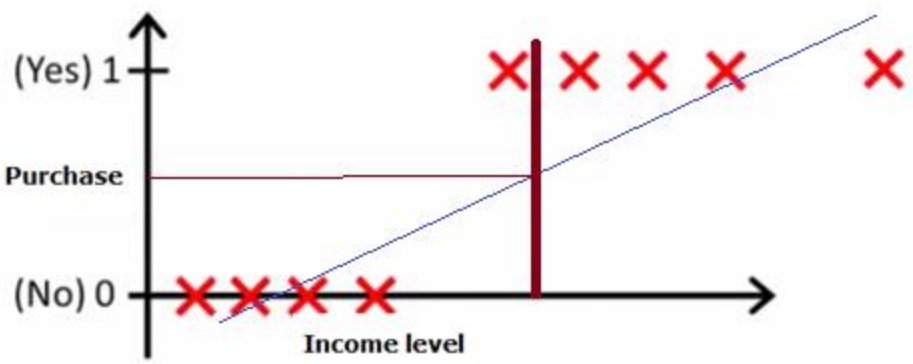


Let us say, we want to model it as a function of income level. One thing we can do is to model using linear regression.



As linear regression out puts a numeric value and not 0 and 1, we can identify a threshold value of y and define such that above that it is 1 and below that is 0. As you can see, it works fine in

the above example. But, imagine a slightly different data set. As you can see linear regression and the same threshold give a poorer result.



In addition, linear regression hypothesis can be much larger than 1 or much smaller than zero and hence thresholding becomes difficult.

In logistic regression we take two steps:

1. Find the estimates of the probabilities of belonging to each class. Case when Y = 0 or 1, the probability of belonging to class 1, P(Y=1)

2. Use a cut-off value on these probabilities in order to classify each case in one of the classes. For example in binary case, a cut-off of 0.5 means that the cases with an estimated probability of P(Y=1) > 0.5 are classified as belonging to class 1, whereas cases with P(Y=0) < 0.5 are classified as belonging to class 0. The cut-off need not be set at 0.5. When the event in the question is a low-probability event, a higher than average cut-off value, although below 0.5 may be sufficient to classify. Deciding a cut-off value is an 'Art' rather than science.

Model

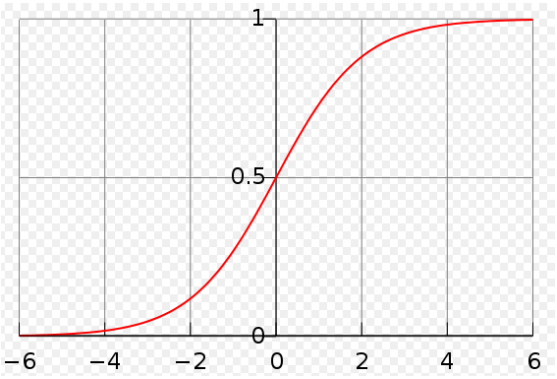
The hypothesis in linear regression is

$$h = \theta^T X$$

However, this can go from -∞ to ∞. So, we modify it into a sigmoid or logistic function as shown below

$$h = \frac{1}{1+e^{-\theta^T X}}$$ *(This is called the logistic response function)*

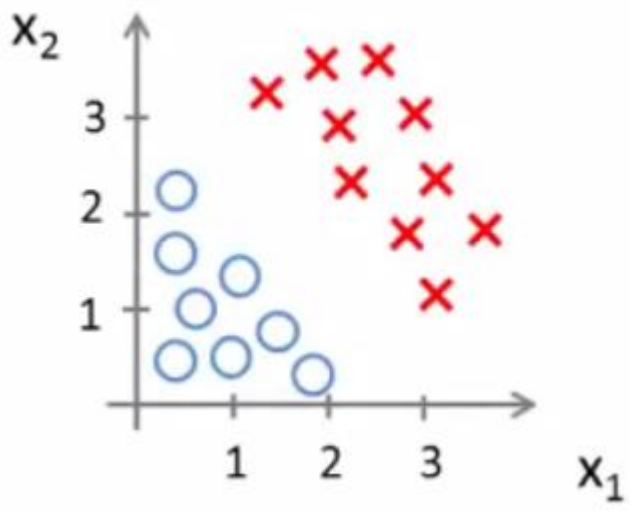
When x= -∞, y_i = 0; When x= ∞, y_i = 1; When x= 0, y= 0.5. Clearly, y always takes a value between 0 and 1 as x varies from -∞ to ∞.



One interprets the value as the probability of y belonging to class 1. So, if the output of logistic regression is 0.7, then the probability that the class is 1 is 0.7

From the above graph, it is evident that if $\theta^T X$ >0, then y>0.5. In a binary classification, it means that the probability that class is 1 is more than 0.5.

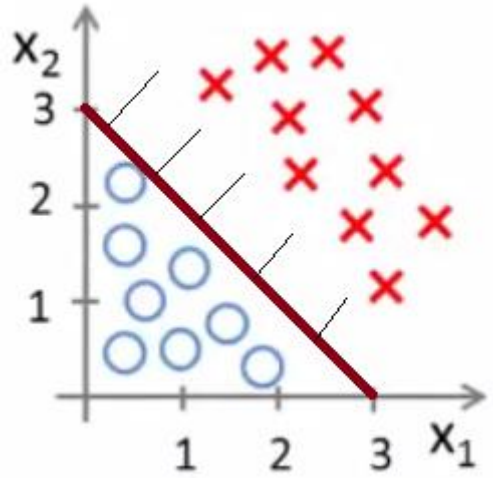
Let us look at the decision boundaries of logistic regression on the data set where the crosses are represented as 1 and circles are 0.



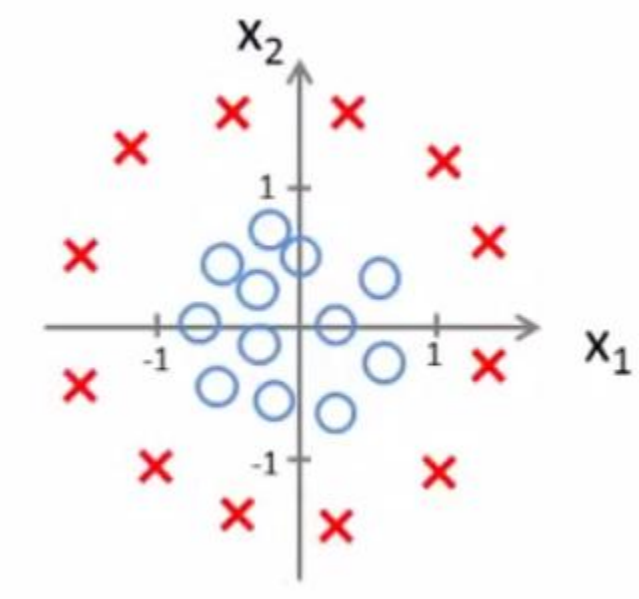
The hypothesis is of the form

$$h = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

g here is the logistic function. Let us say, we compute the parameters of hypothesis and they turn out to be -3, 1, 1. Then as we learned above, we can conclude that the record is 1 (or cross) when $-3+x_1+x_2 > 0$ or $x_1+x_2 > 3$. This is shown in the following figure



For more complex data sets, the decision boundaries have to be more complex.

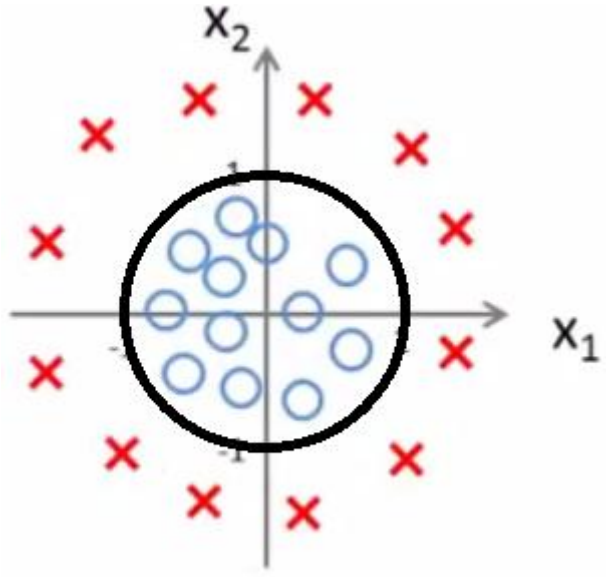


We need to use a quadratic hypothesis space to describe it.

$$h = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 {x_1}^2 + \theta_4 {x_2}^2)$$

Again, let us say, somehow we find that the correct values of the parameters are -1,0,0,1,1.

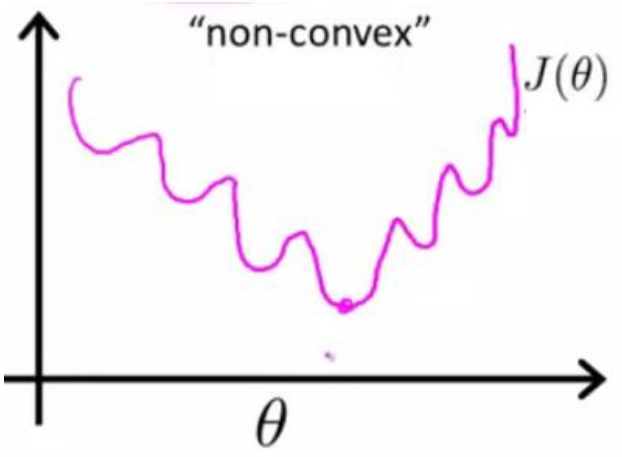
Then, we get the hypothesis that class is 1 (or cross) if $x_1^2 + x_2^2 - 1 > 0$.



So, by going higher order polynomials, we can construct more complex decision boundaries.

Cost function of a logistic regression

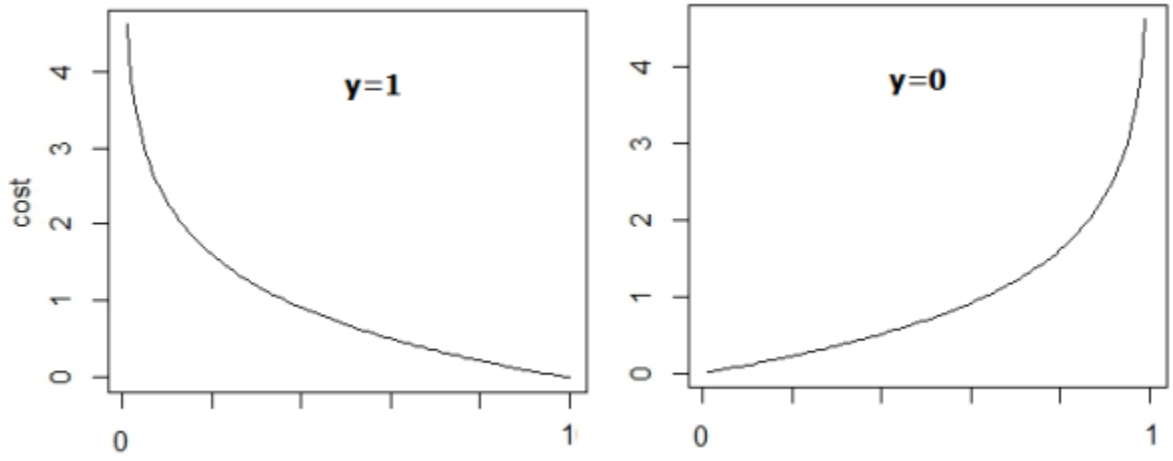
The cost function of linear regression would become very complex if used directly in logistic regression as the hypothesis is of the form of a sigmoid.



Gradient descent would lead to local minimum points and hence cannot be used. We use a different cost function

$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

This looks complex. But, if we plot the cost functions, the intuition becomes clear.



If y=1, and the prediction is 1, then cost =0 (from left hand side). Similarly, cost becomes zero whenever a correct prediction

is made and it becomes very high when incorrect predictions are made.

The above cost function can be simplified as

$$Cost = -ylog(h_\theta) - ((1-y)\log(1-h_\theta))$$

Clearly, this equation reduces to the above cost function based on the value of y.

We use gradient descent to then find the parameters that minimize the cost function.

Analyzing the logistic regression

Let us go back and write our logistic regression equation.

$$p_i = \frac{1}{1+e^{-(\theta_0+\theta_1 x)}}$$

$$1+e^{-(\theta_0+\theta_1 x)} = \frac{1}{p_i}$$

$$e^{-(\theta_0+\theta_1 x)} = \frac{1-p_i}{p_i}$$

$e^{(\theta_0+\theta_1 x)} = \frac{p_i}{1-p_i}$ (this is called the odds of success) Eq(1)

$$\log\left(\frac{p_i}{1-p_i)}\right) = \theta_0 + \theta_1 x \quad \text{Eq(2)}$$

From Eq(2), the log of odds has a linear relationship with the independent variables. From Eq(1), a few things become clear

$$\text{Odds} = e^{\theta_0}.e^{\theta_1 x} \ldots$$

Hence, in logistic regression, the terms are multiplicative and not additive like linear regression. If x increases by 1 and everything remains the same, the odds increase by e^{θ_1}. Another way of saying is that the log of odds increase by θ_1.

Here is some intuition

- Logistic regression models are multiplicative in their inputs.
- The exponent of each coefficient tells you how a unit change in that input variable affects the odds ratio of the response being true.

- Overly large coefficient magnitudes, overly large error bars on the coefficient estimates, and the wrong sign on a coefficient could be indications of correlated inputs.
- Coefficients that tend to infinity could be a sign that an input is perfectly correlated with a subset of your responses. Or put another way, it could be a sign that this input is only really useful on a subset of your data, so perhaps it is time to segment the data.

Case 5.3: Logistic regression example

We will take the example given in http://www.ats.ucla.edu/stat/r/dae/logit.htm. OK! over to R code.

The summary of the model is given as

```
> summary(mylogit)

Call:
glm(formula = admit ~ gre + gpa + rank, family = "binomial",
    data = mydata)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6268  -0.8662  -0.6388   1.1490   2.0790

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.989979   1.139951  -3.500 0.000465 ***
gre          0.002264   0.001094   2.070 0.038465 *
gpa          0.804038   0.331819   2.423 0.015388 *
rank2       -0.675443   0.316490  -2.134 0.032829 *
rank3       -1.340204   0.345306  -3.881 0.000104 ***
rank4       -1.551464   0.417832  -3.713 0.000205 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 458.52  on 394  degrees of freedom
AIC: 470.52

Number of Fisher Scoring iterations: 4
```

In the output above, the first thing we see is the call, this is R reminding us what the model we ran was, what options we specified, etc.

Next we see the deviance residuals, which are a measure of model fit. This part of output shows the distribution of the deviance residuals for individual cases used in the model. Below we discuss how to use summaries of the deviance statistic to assess model fit.

The next part of the output shows the coefficients, their standard errors, the z-statistic (sometimes called a Wald z-statistic), and the associated p-values. Both gre and gpa are statistically significant, as are the three terms for rank. The logistic regression coefficients give the change in the log odds of the outcome for a one unit increase in the predictor variable.

- For every one unit change in gre, the log odds of admission (versus non-admission) increases by 0.002.
- For a one unit increase in gpa, the log odds of being admitted to graduate school increases by 0.804.
- The indicator variables for rank have a slightly different interpretation. For example, having attended an undergraduate institution with rank of 2, versus an institution with a rank of 1, decreases the log odds of admission by -0.675.