**You can skip until page 9 for the Data Prediction Information.**

# The Power of Math 2

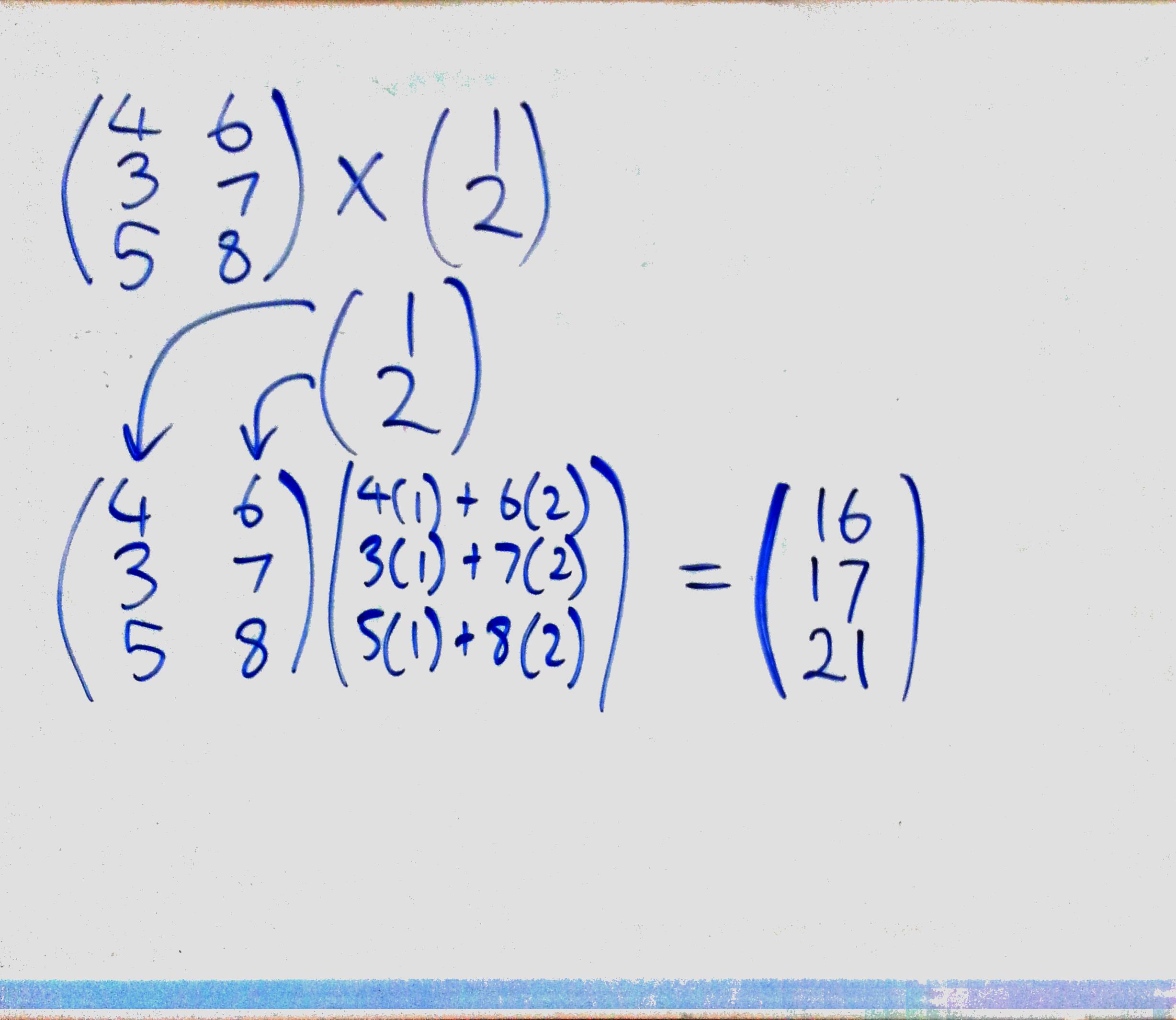
In addition to the mathematics we used in the MSIR Model, we now need to know a bit more! I’ll explain it as easily as possible.

First, we need to know Matrix Multiplication, in which we discussed in the first model.

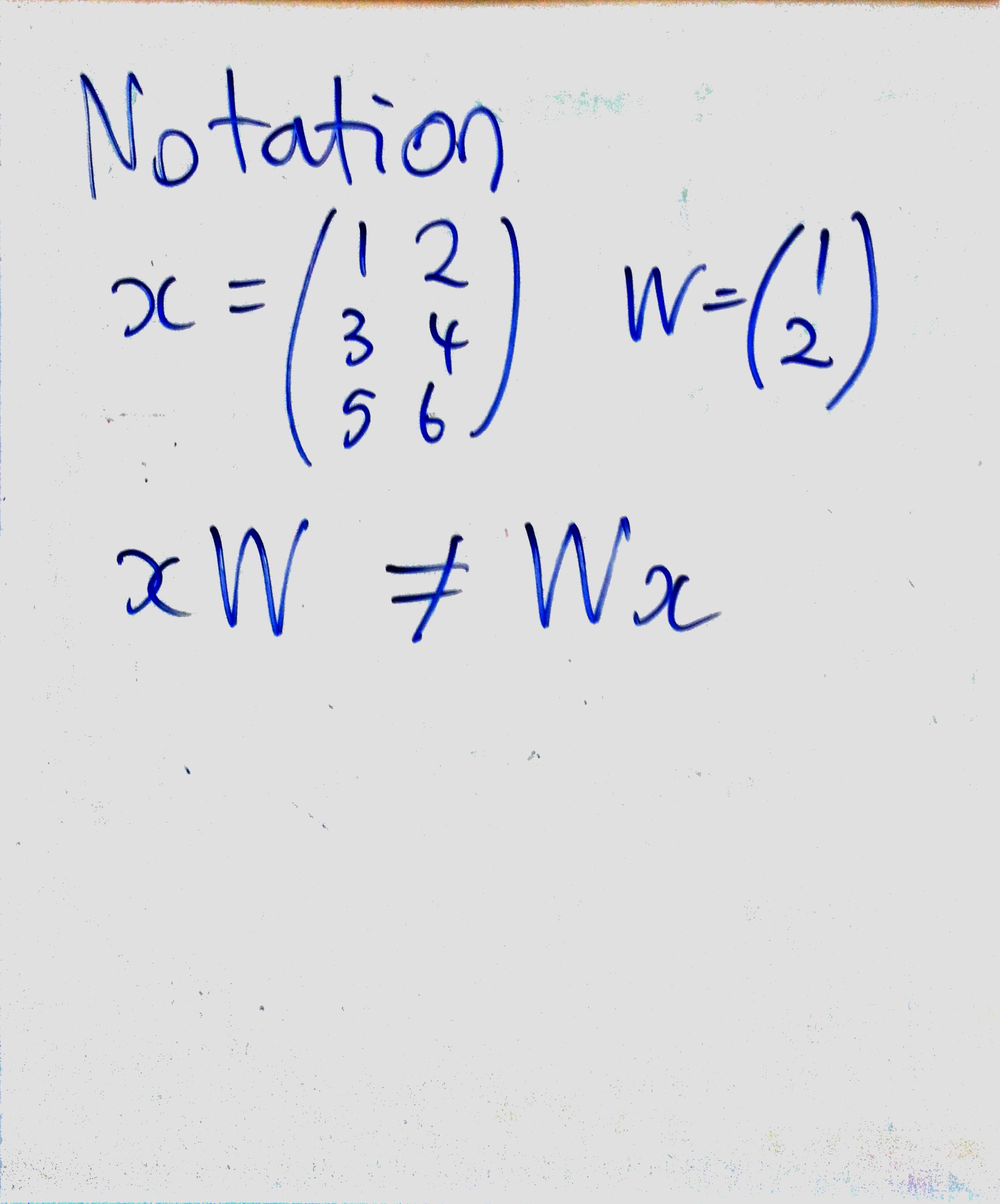
Remember

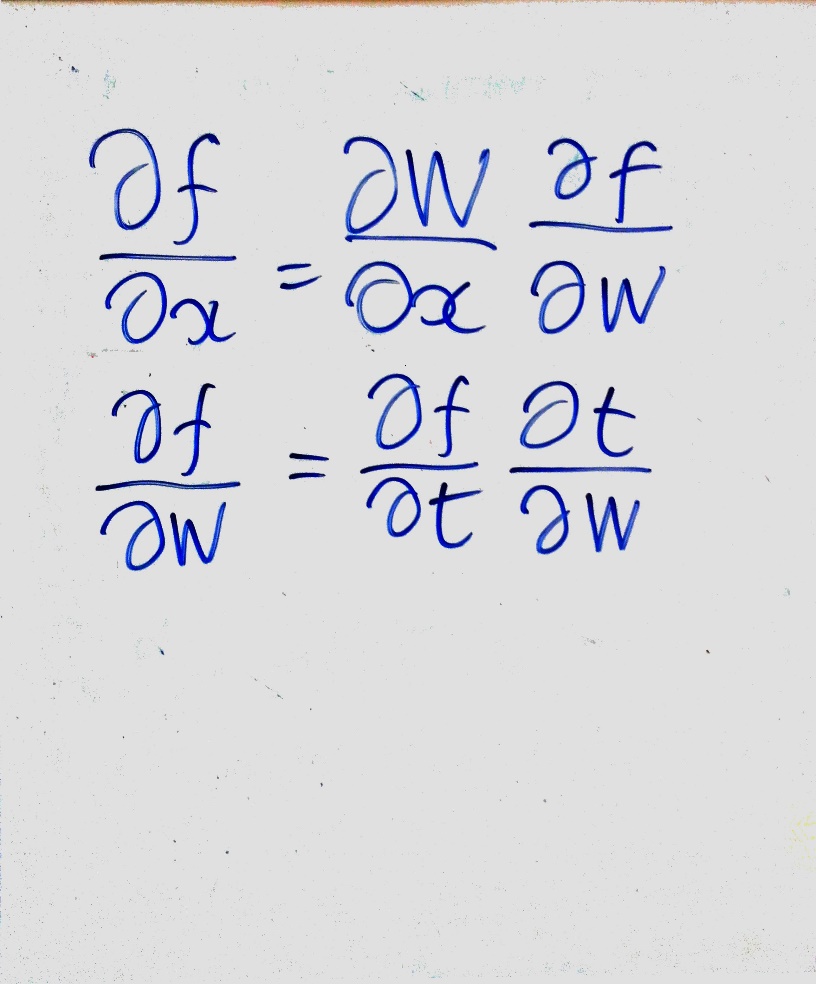
Now, we extend the idea to include other matrix sizes.

Like this:



Now, using mathematical notation, we can rewrite the above into:



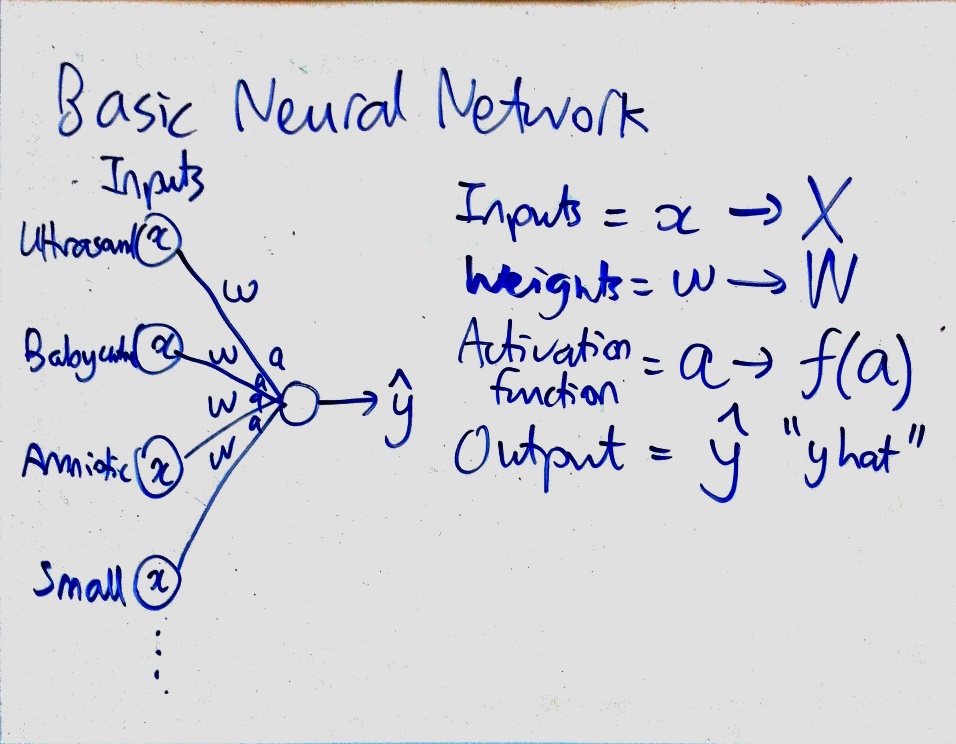
Our OCD model also uses calculus. I’ll just introduce the chain rule we used, but it isn’t necessary to understand:

This just means when we differentiate (or find the slope) of some function (or equation), we can separate the task of doing so into smaller parts.

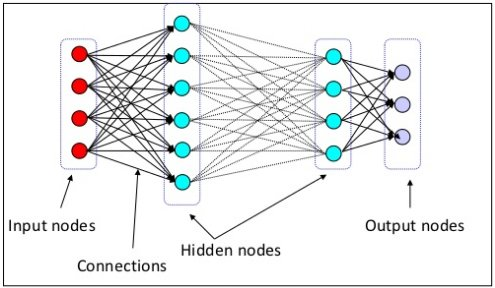
Then, we can combine these smaller tasks together (by multiplying them), and then we can get the differential to some very complicated function!

# Deep Learning Model

Now, I’ll need to introduce you to the basic concepts of how we got our results. We used a Deep Neural Network Model with 5-6 Hidden layers, and used the normal Gradient Descent Optimisation Method. We also used for our sigmoid, RE-LU, ELU and Tanh activation functions.

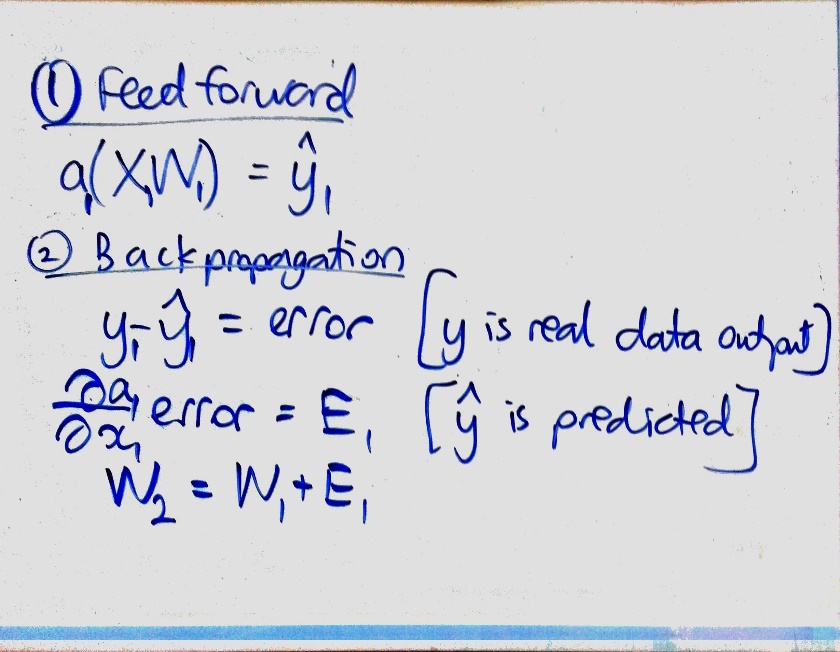
A basic Neural Network or NN for short is to the right:

A NN is modelled on the workings of a human brain. Neurons are connected via Axons and Dendrites and the Neurotransmitters also cause the electrical signals to pass through an activation function. A NN tries to mimic the behaviour of a human brain.



As you can see, the inputs are each feature. These features are the search terms (like Babycentre, Amniotic Fluid). These search terms are then combined into one large dataset, in which we will name X.

The weights or W is the measure of “importance” for each feature. These thresholds cause the features to either be used or not used, but at a proportional level. For example, if a Node (the lines connecting to a large Circle – the Neuron) produces a weight of 20%, then the feature data which is connected to the node will be weighted by 20%. Other data connected to other nodes will be weighted differently (this difference is ascertained during the training phase of the model).

Now, to ascertain these mysterious weights, we first initialised the weights with a Normal Distribution (we used a Normal Distribution, as it most accurately models real world phenomena).

Then, we used some mathematical functions:

Feedforward is the process of just using the random initialised weights and multiplying it with the input matrix X (which has all the features).

Then, we pass it through an Activation Function, like a sigmoid function. It looks like this:

An Activation Function tells the weight to be accepted or not, and this acts as a threshold for the entire model.

This then produces y-hat (or the predicted value). We then use calculus to minimise the loss function (or the discrepancy between y-hat and the real value y).

This minimisation method was aided with the Adam Gradient Descent method.

Then, we use the error value, and input it into the differential equation derived from the activation function (or sigmoid).

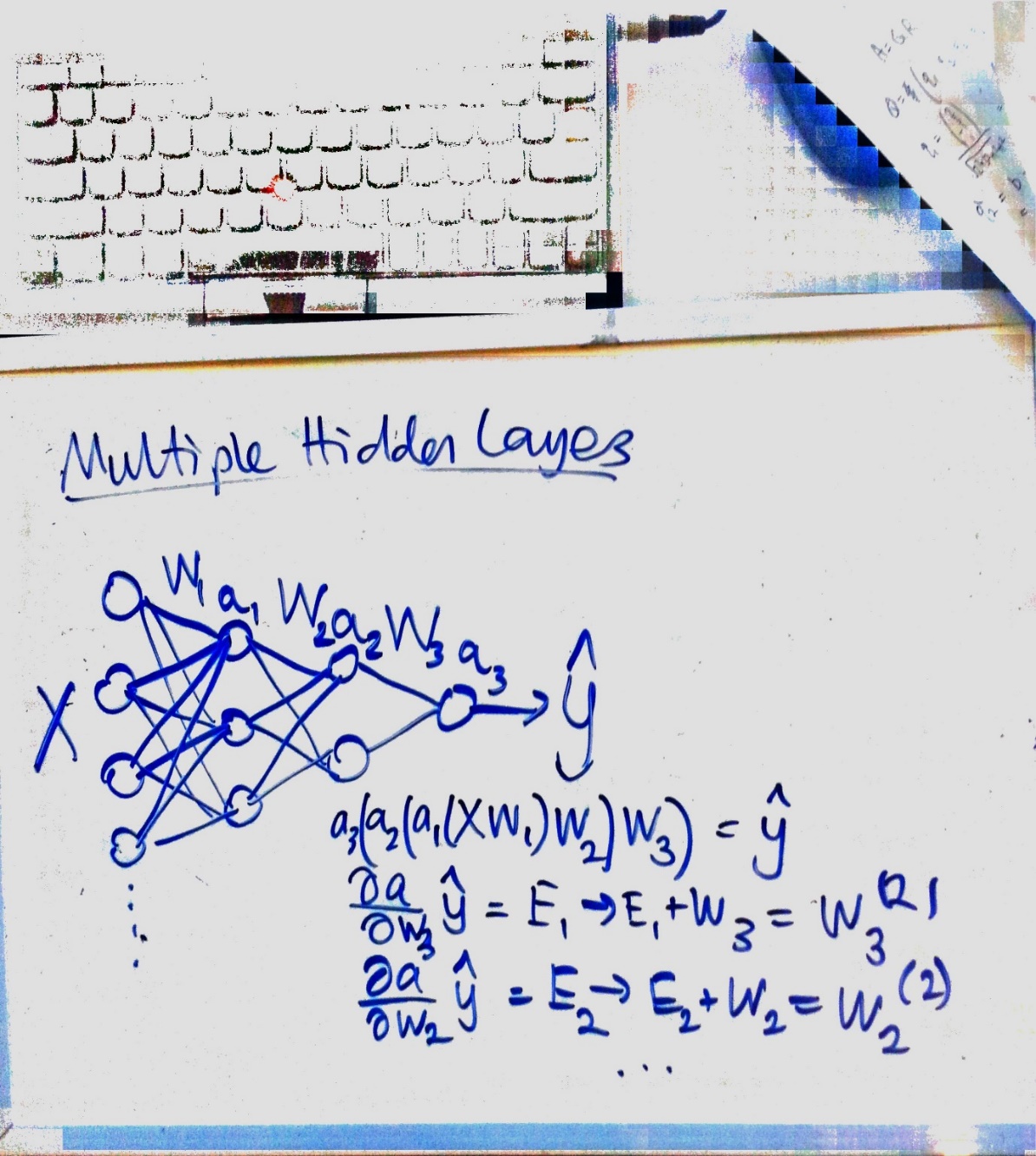
The sigmoid’s derivative is:

After that, we use the output value of the final “error” correcting value derived from inputting the error value into the differential function, and we update the original Weight matrix by it.

To combine the functions, we get:

After say 200 training cycles, the weight matrix can accurately model the “importance” of each feature.

Now, our Deep Learning Models use more than one Weight and more than one Layer.



As you can see, we used multiple hidden layers. The process to ascertain the weights was the same as before, however, we need to do extra steps to get each layer’s weight matrix W(n).

We used some calculus chain rule magic, and multiple layer weight matrices were produced. As you can see in the diagram, the Activation Function applied onto the Weight.

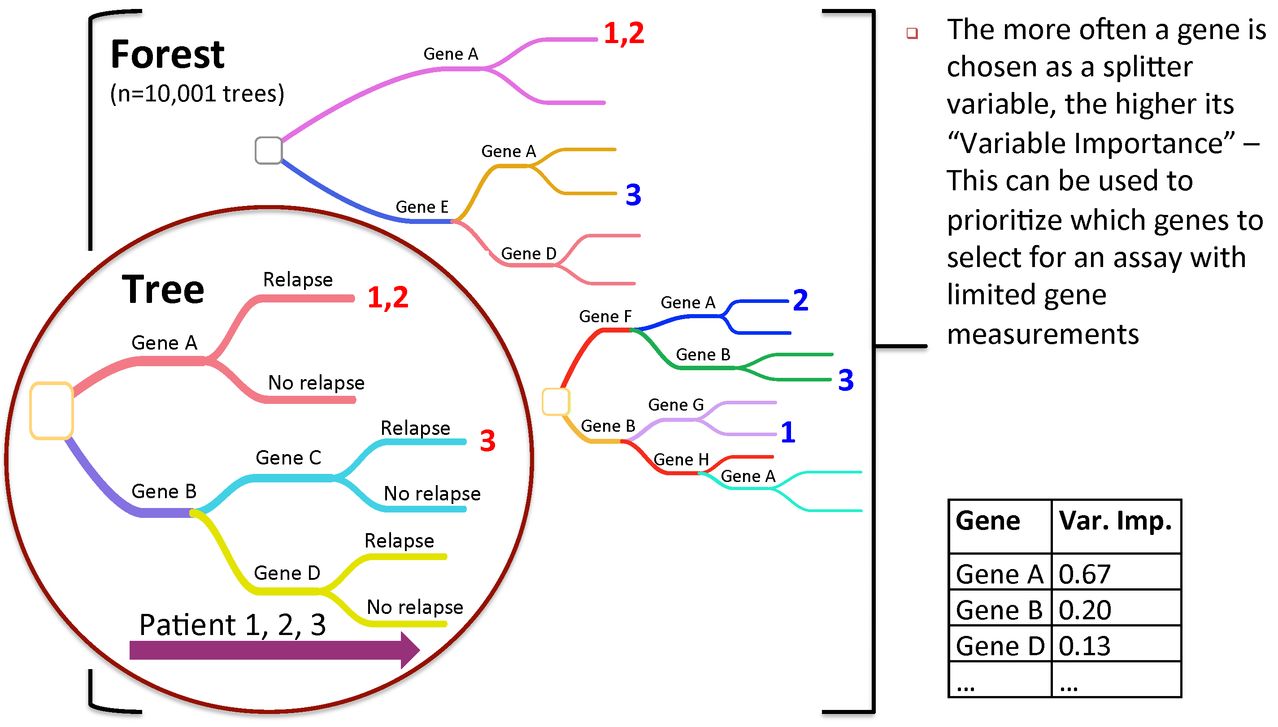
So our final mathematical model was:

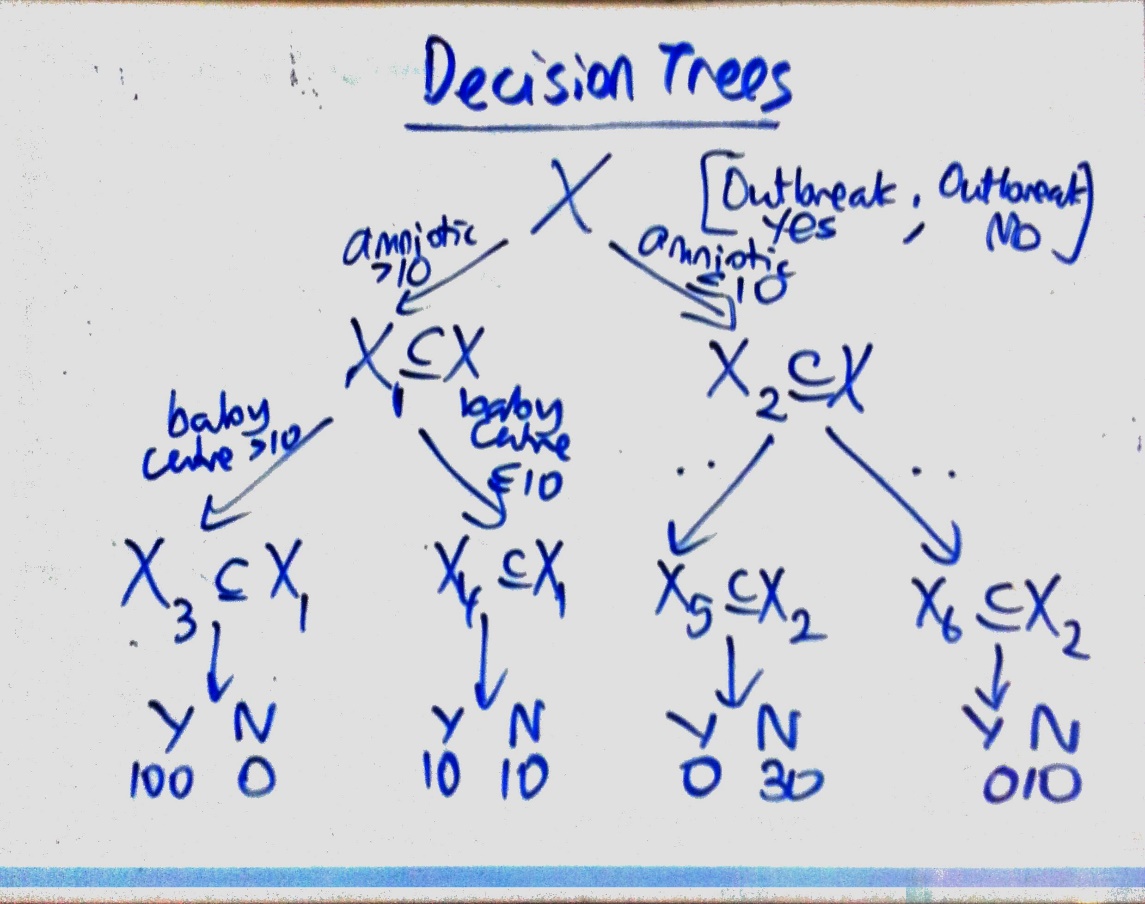
# Random Forest Model

The other models we used were Random Forests.

These are “opposite” to what a Deep Learning network does. I won’t explain a lot of how it works, but I’ll try to tell you the main basics.

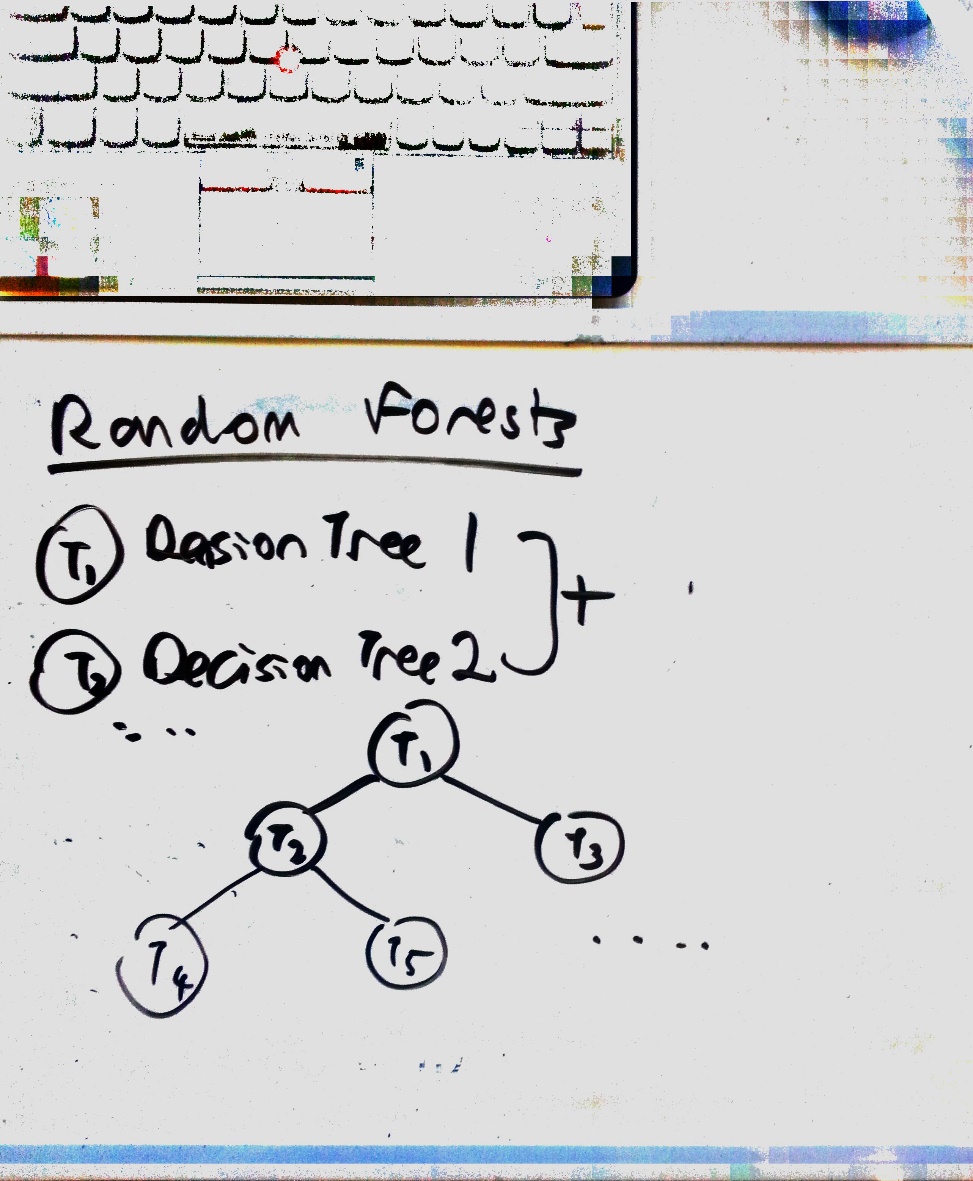
A Random Forest or RF for short, is a collection of many Decision Trees or DT for short. DTs are like a classification tree. If you used a classification book for let’s say tree types or leave types, a DT is the same. (Like 1. If leave is wrinkly 🡪 go to 2.)

An example is below:



A DT subsets the data using the features (or search terms), to find which ones are more strongly related to whether there is or there isn’t an outbreak.

Now, RFs (Random Forests) are a collection of multiple DTs.



Each DT acts as a filter within each branch of the RF, and thus increases the accuracy of the model. We used approx. 1000-2000 DTs for each Random Forest model.

# Data Prediction

Now, the process to get our output of a 1 for “YES Outbreak” and 0 for “NO Outbreak” is detailed into steps, including:

1. **Data Collection**
2. **Backwards Extrapolation and Regression**
3. **Binary Conversion and Continuous Conversion**
4. **Model Combination**
5. **Over-prediction and Under-prediction**

## **1 Data Collection**

So, first, we needed to know how many microcephaly cases there were in each state. We used the UN/PAHO resources in conjunction with the CDC resources to ascertain a Zika dataset.

This dataset is publicly available through:

<https://github.com/BuzzFeedNews/zika-data>

and <https://www.kaggle.com/cdc/zika-virus-epidemic>

The dataset looks like this:



As you can, we can subset the states we want (in this case Bahia) by using Excel’s powerful AUTOFILTER method. Excel is just so good!

So for each training model (we had 4 for each language – so 8 in total (2 languages) ), we did the same method of subsetting. However, I’m just going to demonstrate what we did for one state – Bahia (as it gets repetitive).

Using Excel’s graphing functions, we graphed the microcephaly cases for Bahia, as seen below:

## **2 Backwards Extrapolation and Regression**

Now, after we have a basic understanding of the trend of Microcephaly cases, we extended the trend backwards and forwards.

But how do we do that? We used Excel’s curve fitting functions.

1. **Linear Regression (y=a+bx)**

So we note down:

1. **Quadratic Regression (y=ax^2+bx+c)**

So we note:

1. **Exponential Regression (y=ae^bx)**

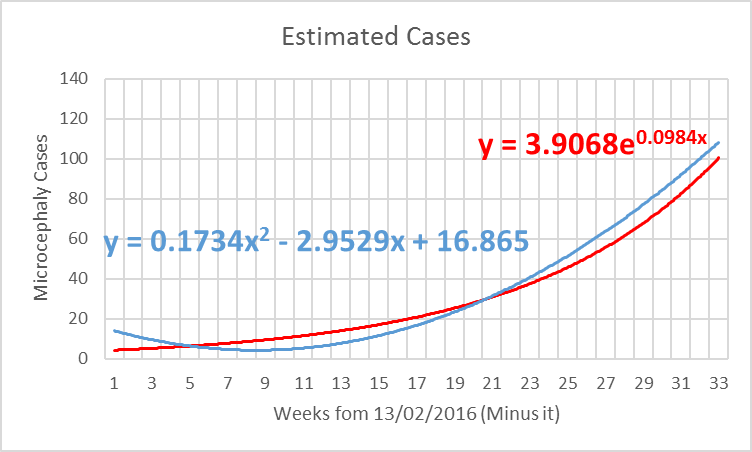
So we note:

Now, using these 3 regression trends, we output an extrapolated trend:

Clearly, the quadratic one is not very good at all (especially when it is -67 weeks from 13/02/2017 (or around 13/11/2014).

Thus, we try to rescale it.

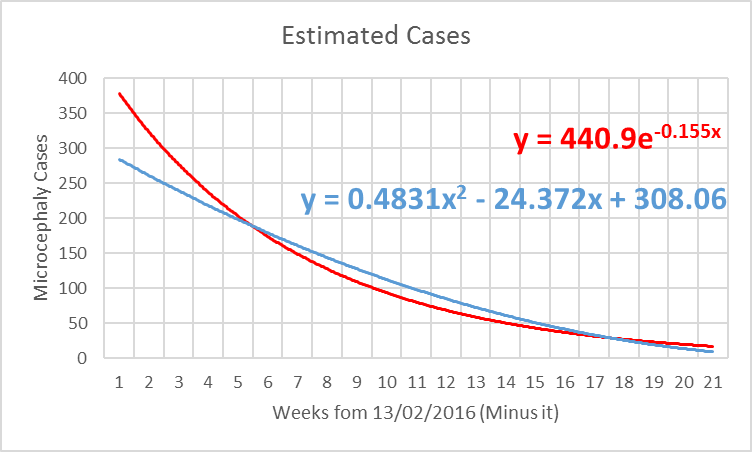
So, for the backwards regression, after some many trials of scaling of each estimator, we get:



Or in formula terms:

For backwards extrapolation.

Now, we also needed to do forwards extrapolation.



We assumed the amount of microcephaly cases were decreasing over time from the initial epidemic report.

Or in formula terms:

For forwards extrapolation.

Thus, when we combine them, we get:



In maths:

## **3 Binary Conversion and Continuous Conversion**

Now, we had two choices to select from:

1. Either use a binary system (so 1 for outbreak, 0 for not outbreak)
2. Or: a continuous system (no. of microcephaly cases)

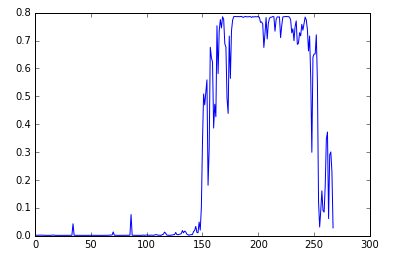
We considered both in fact, and you can just change the model’s internal settings by using the continuous or binary datasets.

But, in the end, we decided that the binary system should be our main indicator, as it aligns with the competitions guidelines (where they want to know when there is an outbreak or not (so 1 or 0)).

However, we did train the model on a continuous case, then outputted into a binary case through normalisation and scaling. So in essence, we used both binary and continuous cases in our models.

In the code, we submitted, we showed the graphs for the binary/continuous classification:





## **4 Model Combination**

So after the DL and RF models were trained on Spanish and Portuguese Search Terms, including:

|  |  |
| --- | --- |
| **Portuguese** | **Spanish** |
| ultrassom,pequeno,cabeça,  trimestre,grávida, criança  mãe,Mamãe, bebê,médico  babycenter,clínica,anormal,  Tamanho,circunferência  cérebro,nervo,tumor,sangue,  amniótico  fluido,nascermos,gravidez,  Fralda,Vômito, nascimento  hospital,incapacidade,Desativado, crescimento | Ultrasonido,pequeña,  cabeza, trimestre, embarazada  Niño, madre, mamá, bebé, doctor  Babycenter, clínica, anormal, tamaño, circunferencia  Cerebro, nervio, tumor, sangre, amniotico  Fluido, nacido, embarazo, panal, Vomito  Nacimiento, hospital, discapacidad, discapacidad, crecimiento |

We combined the models!

(Also, we chose Spanish and Portuguese because:

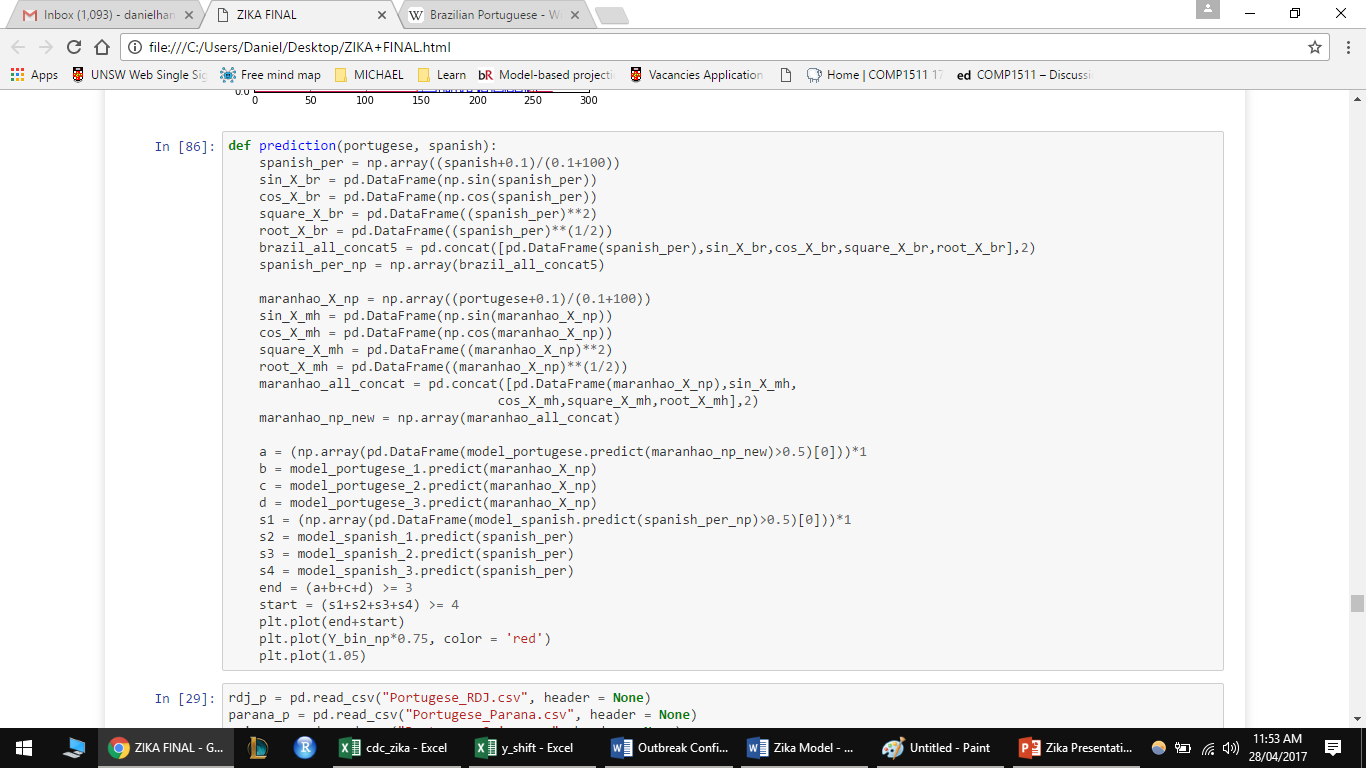
1. Portuguese was the main national language of Brazil
2. Spanish was also a main secondary language of Brazil )

So, after we plotted each state (tested approximately 7-9 states) for each model, we realised that Portuguese search terms were in fact superior to Spanish Search Terms.

We checked Google why, and we found the number of people speaking Portuguese in Brazil was around 200 million, compared with Spanish’s 6 million only.

Thus, using this, we scaled the Portuguese models in favour of the Spanish Model (as the data points are less skewed – since more people use Portuguese search terms).

Thus, our code to show this is below:



As you can see, we scaled Portuguese by a factor of 3 (less is better), whilst Spanish was by 4 (more is worse).

Thus, we ended up with:

Now, you might be wondering where I got these numbers 3 and 4 from.

Since we used 3 RF models and 1 DL model for each language, we scaled per how many models were used for each.

So, that means out of the 4 models, Spanish took 3 importance whilst Portuguese took 4.

In other words, Spanish was ¾ or 75% useful, whilst Portuguese was 4/4 or 100% useful.

In the end, we get:



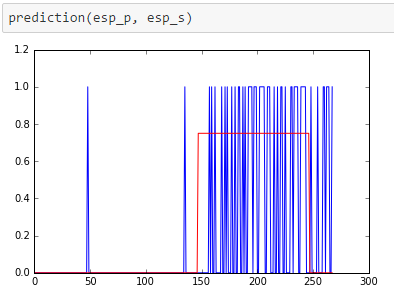
Where red indicates the Projected trend of the Zika Outbreak. The starting is from November 2014 and the Ending is November 2016.

The blue lines indicate if there is an outbreak, where if you see one, it means there is one.

## **5 Over Prediction and Under Prediction**

Now, sometimes our predictions were over or under the projected trend.

Like so:



Now, we could correct these issues with:

1. More data (Need a bit more time to train the model)
2. More search terms (We used 20. We could do more)
3. More training cycles (we used 200-300. I wanted to do 10000 or more, but that’ll destroy my school laptop.)

Or, we should have investigated why these outliers existed.

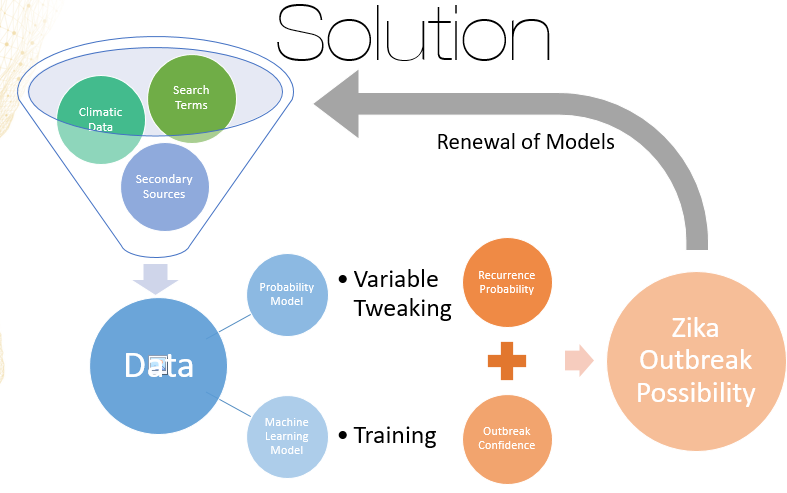
# Conclusion and Ambitions

The team really wanted to make our MSIR model a reality if we had the readily available data.

If we could do that, the model could be used in other areas of disease monitoring like the Common Cold.

With MSIR acting as a validator, and the OCD model acting as a predictor, both can be combined to further produce valid and accurate predictions.

Thus, the MSIR and OCD models complement each other, and if only one is used by itself, then the true power of our models can never be truly achieved.



Anyways thanks everyone for reading! The competition was fun and exciting! I also learnt so much! Thanks for organising this.

# Formulas