Muhamed Essam

September 1, 2018

Road to Deep Learning

Chapter 1

Basics

- Intro to Neural Networks
- Biological Neural Networks
- ANN
- MLP
- Activation functions
- Optimization
- Gradient Descent
- First Model with Keras

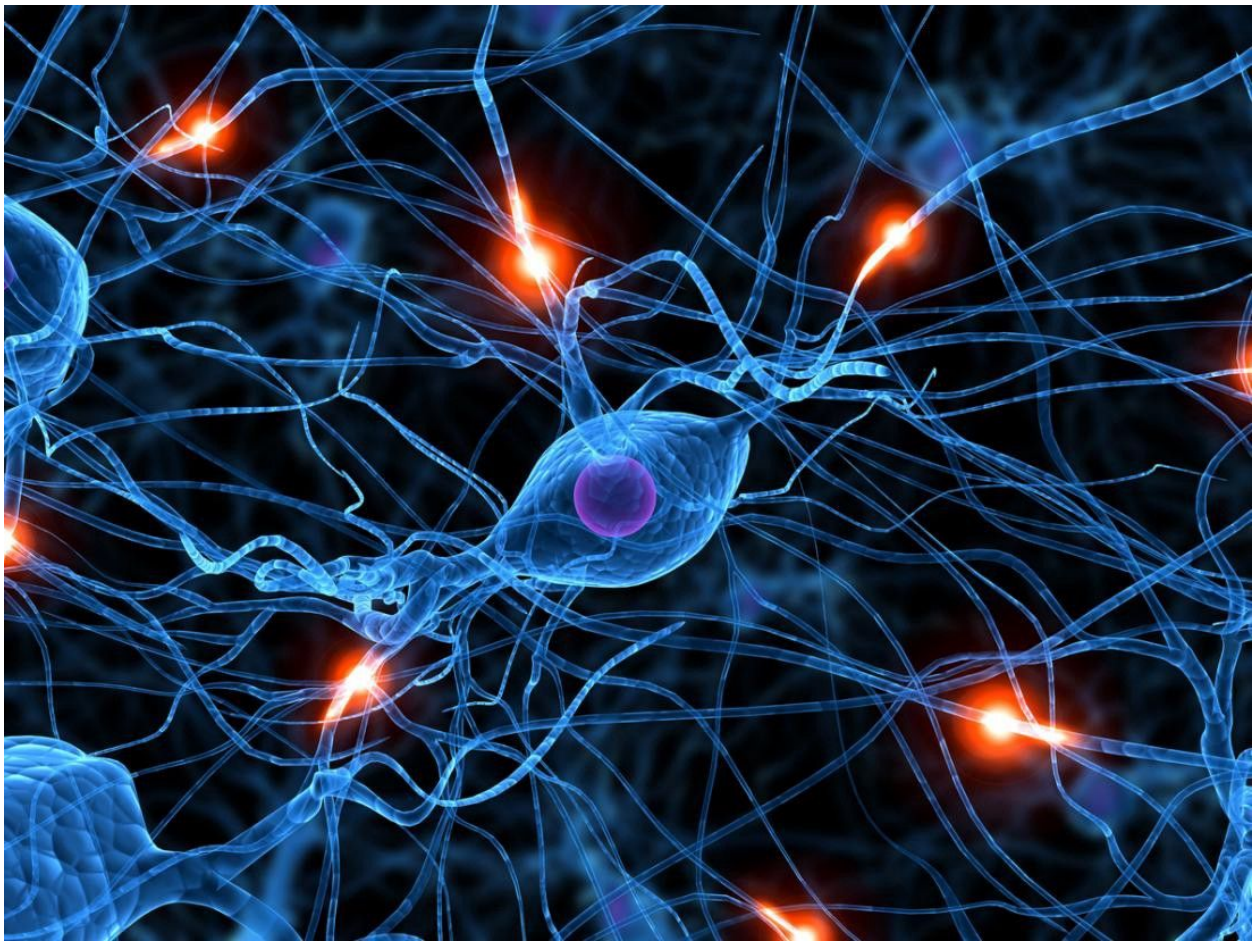
Intro to Neural Networks

لو انت Developer فأكيد زهقت من bugs في Code بتاعك من ان في مشاكل كتيرة أوي بتقابلك عشان تطلع حاجة تناسب Client وممكن تدخل نفسك في لسته من conditions لمجرد بس انك خايف من أي Action هوا ياخده وتبقي انت مش متوقعه .. وده بيبدأ يوصلنا لسؤال مهم .. هو مفيش طريق غير ده ؟ مينفعش أوصل لأسلوب أخلي الكود بتاعي يفكر ؟ وياخد قرار ؟؟

- عمرك سألت نفسك , انت ازاي بتفهم أي معلومة ؟ أو إزاي دماغك بتشتغل ؟
- تفتكر لو خيّرتك انك تبدل دماغك بكومبيوتر هتقبل ؟
- طيب تفتكر مين أفضل Computer ولا human brain ؟
- تفتكر ازاي المعلومات في دماغك مش بتخش في بعضها؟

Biological Neural Networks

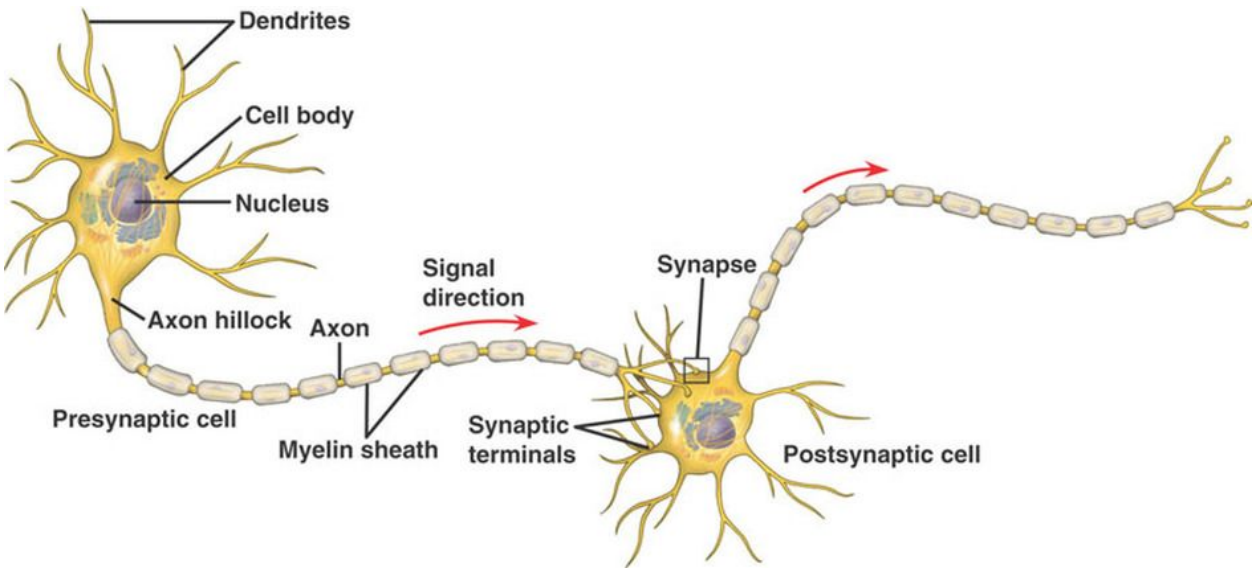
لان احنا محتاجين نعمل نظام مشابه لطريقة تفكير الانسان فاحنا كان لازم ندرس ازاي المخ البشري بيتعامل مع المعلومات .



تعالي نحاول نفهم الدنيا بطريقة بسيطة في الحقيقة المخ بيتكون من خلايا عصبية بنسميها Neuron وهي في الحقيقة بتستقبل المعلومة من الحواس بتاعتنا زي العين أو الأذن ...

الخلايا العصبية دي بتبقي مترتبة و متوصلة ببعض باشكال مختلفة وبتكون شبكة بنسميها الشبكة العصبية.

في الشكل اللي تحت ده هتلاقي فيه رسمة Neuron من جوا وهي متوصلة ب Neuron تانية .



أهم النقط في الشكل ده بالنسبة لنا هي ان قدامك راس الخلية وهو اللون الأصفر الكبير ده وبنسميه Soma وليه ديل أو طرف وهو اسمه Axon ... محتاجين ناخد بالنا من ايه بقي ؟

ان احنا بيبقي عندنا وليكن information في neuron وعايزة تبدأ تنقلها ف axon بينتهي ب Axon terminals أو بنسميها Synapses وبتستقبلها أطراف ال neuron التانية عن طريق dendrites وتوصلها لجوة Soma عند Nucleus نفسها وهكذا تبدأ هي كمان توصل اللي وراها بنفس الطريقة ..

طيب ايه النقطة المهمة بالنسبة لنا ؟ ان في الحقيقة كل neuron متوصلة بالاف من neurons فانت لو فكرت هتقول ان في فوضي ممكن تحصل والمعلومات تخش في بعض ... بس في الحقيقة بيحصل نقطة مبهرة جدا وهي ان كل neuron بتتحكم في جيرانها اللي متوصلة بيهم ياما تعملهم firing يعني تشغلهم أو لأ ! فبالتالي neurons اللي شايلة المعلومة المطلوبة بس هي اللي بتوصل..

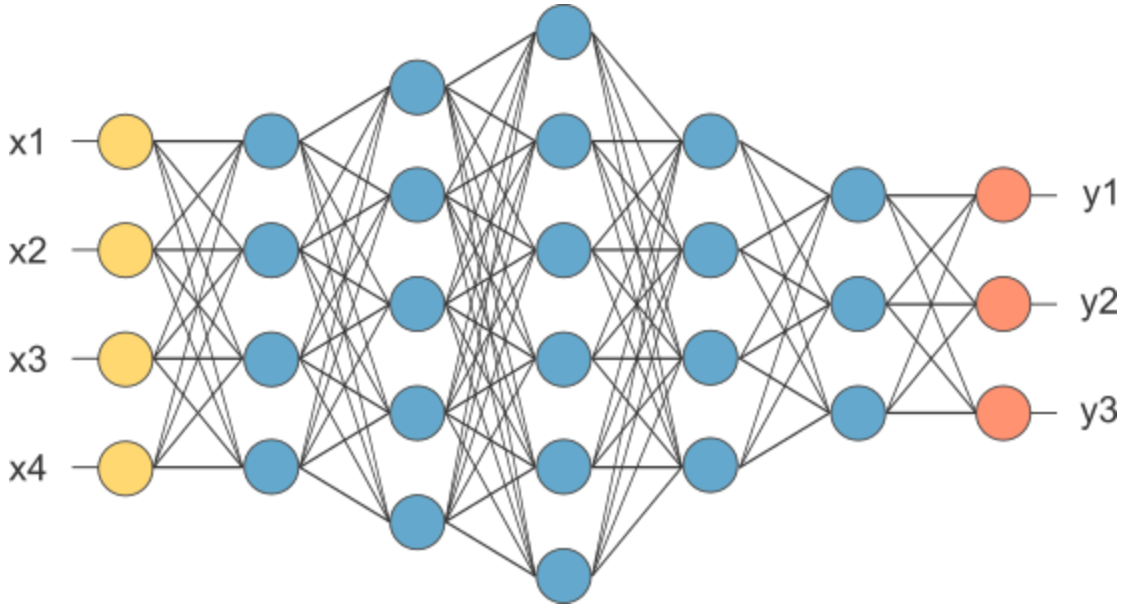
نوضح كلامنا مرة تانية كل اللي قولناه :-

- المخ بيتكون من شوية Neurons متوصلة ببعض بتنقل المعلومات .
- كل Neuron بتبقي متوصلة بالاف من neurons وبتعمل activation / firing لل neurons اللي بتأثر معاها في المعلومة
- المعلومات دي بتتمثل في الاخر ب image أو memory معينة أو... الخ

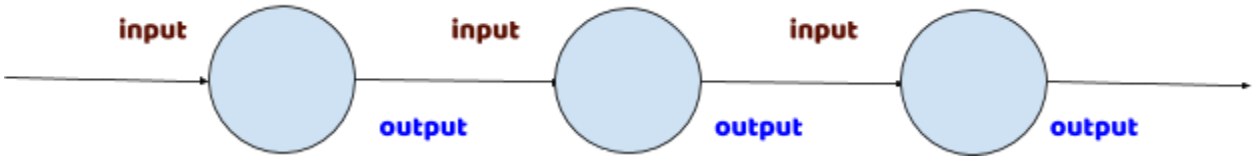
المفروض دلوقتي ان انت بدأت تربط انا احنا هنحاول نقلد الفكرة دي بشكل اصطناعي Artificial ونشوف هنقدر نوصل منها لايه !

Welcome to Artificial Neural Networks

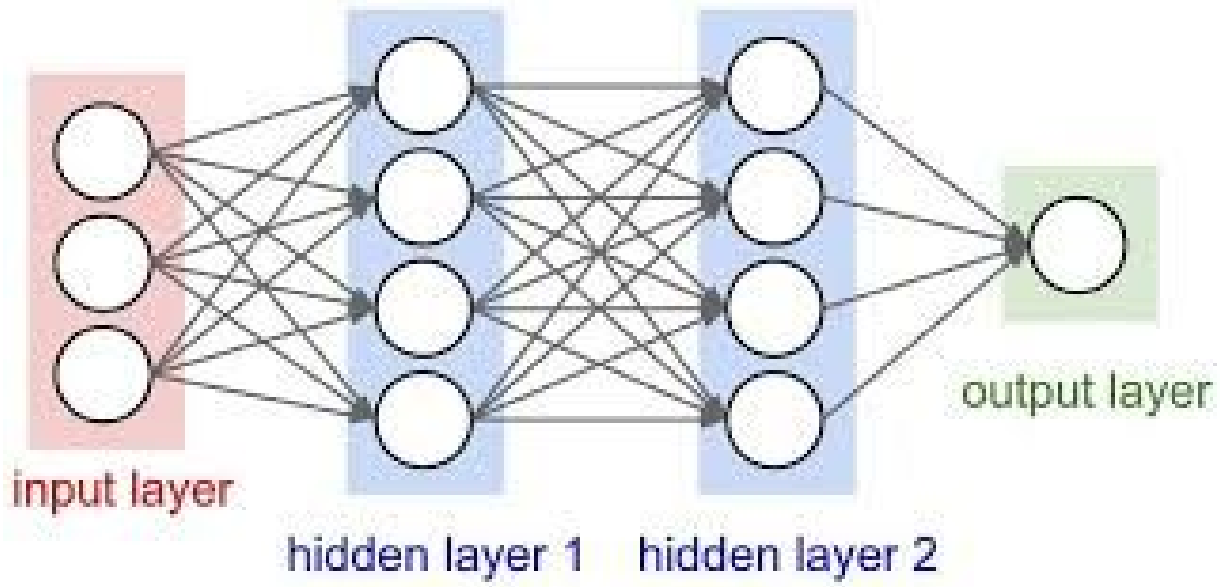
من هنا هنبدأ ندخل مجالنا وهو Computer Science فاحنا دلوقتي في خطوة ان احنا نقلد فكرة عمل neurons بتاعت المخ تمام ؟



الكرات الزرقا والحمرا دي بالنسبة لنا هي Neurons أو ممكن تسميها Nodes طيب احنا برده هنا عايزين نوصل Neurons ببعض زي ما اتكلمنا في human brain عن , Dendrites Synapses هنا احنا هنوصل Neurons ببعض بحيث ان((كل Neuron ليها input وهو output اللي قبلها وليها output وهو input اللي بعدها)) ! ..... * بص تحت علي الرسمة و اقراها مرة تانية*



تمام زي ماقلنا في human brain ان neuron متوصلة بعدد كبير جدا من neurons فاحنا برده هنا هنوصل كل neuron باكتر من neuron .. بالمنظر ده



دي في الحقيقة شكل Basic لل Artifcial Neural = ANN Network ملاحظ ايه ؟

- في 3 حاجات رئيسية وهي , Input layer , Output layer hidden layer
- كل layer متكونة من عدد من Neurons

- كل Neuron متوصلة ب كل Neurons اللي وراها واللي قبلها ... وده اللي بنسميه Fully Connected Neural Network .
- في neuron واحدة في output layet يعني Single output طبعا ممكن يكون عندك Multiple outputs علي حسب Task

في سؤال مهم ؟ ازاي neural network بتتعلم ؟

طب هرد عليك بسؤال أهم وهو أزاي انت نفسك إتعلمت ؟؟

. في الحقيقة انت في فترة كبيرة من حياتك مشيت فيها بمبدأ Supervised Learning يعني بتشوف حاجة قدامك حد بيعملها

وانت بتقلده لحد ما تقدر تعملها

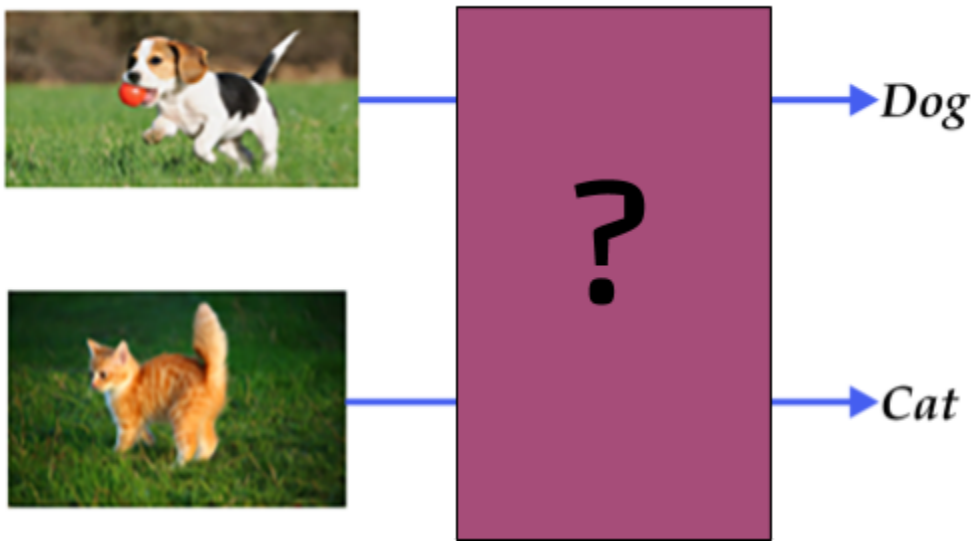
في أمثلة كتيرة أوي لكدة لو فاكر مثلا انت ازاي اتعلمت تميز مابين الحيوانات كنت مثلا بتشوف صور ليها و يقعدوا قلولك دي قطة دي قطة مثلا ده كلب وبعد كده يبدأوا يسألوك ده ايه ؟ فانت تقول قطة فيقولك صح .. فانت عرفت منين ؟ او اتعلمت منين ؟ من تجارب سابقة ليك!

طيب وصلت لايه من الكلام ده ؟

لان احنا بنحاول نقلد طريقة تعليمنا احنا كبشر فاحنا هنبدأ نستخدم أسلوب مشابه في Neural Networks !

فاحنا هنبدأ نوفر شوية Data ولو تخيلنا ان احنا معانا Neural Network model ده وأكنه Black Box يعني حاجة انا مش عارف شغالة ازاي ... انا بس بوفرلها Data عبارة عن شوية Inputs وال Outputs بتاعتها

بالمنظر ده

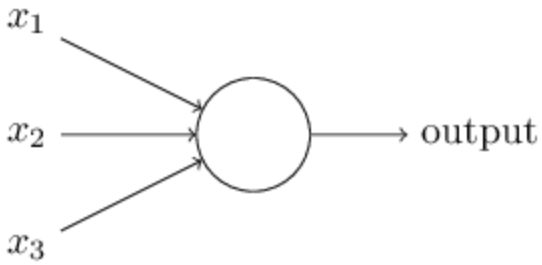


فانت دلوقتي معاك Model بيبدأ يعملك Classification يعني بتدخله صورة ويقولك هل ده قطة ولا كلب ! ال Model بالنسبة لنا Black box يعني انت بتتعامل معاه دلوقتي انك بتديله inputs , outputs ..... وهو هيبدأ عن طريق Data دي يقدر يتعلم ولو سألته علي حاجة بعد كده هل هي قطة أو كلب يقدر يقولك توقعه أو تصنيفه ..

يبقي نقدر نقول ان Neural Network شرط انها تتعلم انك توفرلها Labeled Data يعني data ليها output سليم بنسميه Label .

بس في الحقيقة عشان Neural Network تتعلم بتحتاج تمشي علي مرحلتين أول حاجة هي Forward Propagation وتاني حاجة هي Backward Propagation .. بس قبل مانوضح معناهم ايه وبيتموا ازاي تعالوا نبدأ ب أساسيات Neural Network !

زي ما ال basic structure بتاع اي جسم عندك هو ال ذرة عندنا هنا في Neural Network ابسط وحدة هي Perceptron



ده شكل Perceptron بيبقي ليه أكتر من input اللي هما X1 , X2 , x3

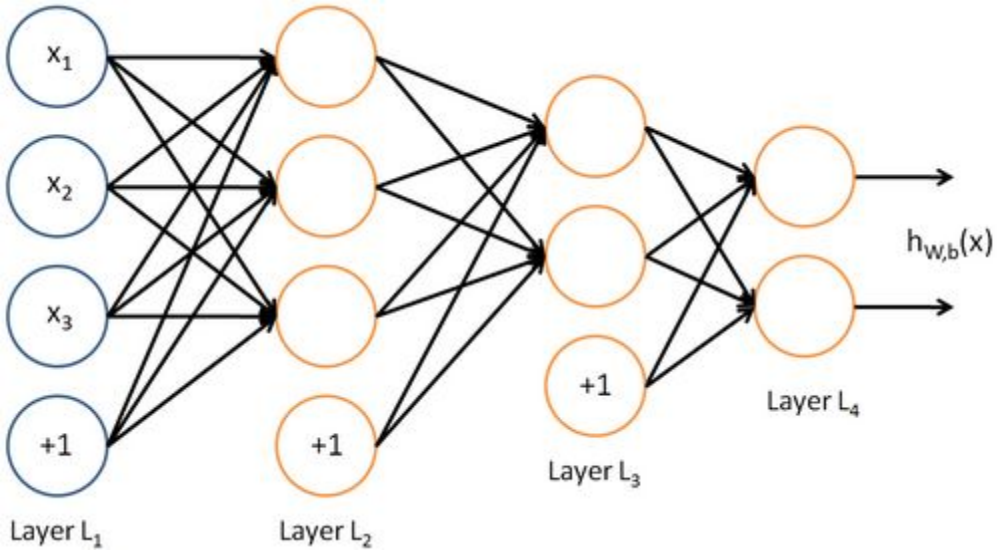
فال Perceptron بتقوم بعملية جمع ليهم وتطلعلك Output بس في Trick ان دلوقتي ده لو System عندك تفتكر هل كل Inputs بتأثر في Output بنفس القيمة ؟ أكيد لأ

كل واحد ليه أهميته .. مثال يعني لو بنميز ما بين راجل وست هل لون البشرة بيأثر زي مثلا طول الشعر ؟ أكيد لأ كل حاجة

ليها عندك أهمية بنسميها احنا هنا Weight وعشان كده بنبدأ نضرب كل Input اللي هي X عندنا في أهميته أو Weight يعني اللي هنسميه W

طيب يبقي كده وصلنا ان y =w1*x1 +w2* x2+w3*x3

فاضل خطوة بسيطة جدا وهي ان احنا بنعمل Adding لل Bias وهو بالنسبة لنا Free term يعني هو مش بيبقي ليه variable input ولا حتي Connected بال Previous Layer

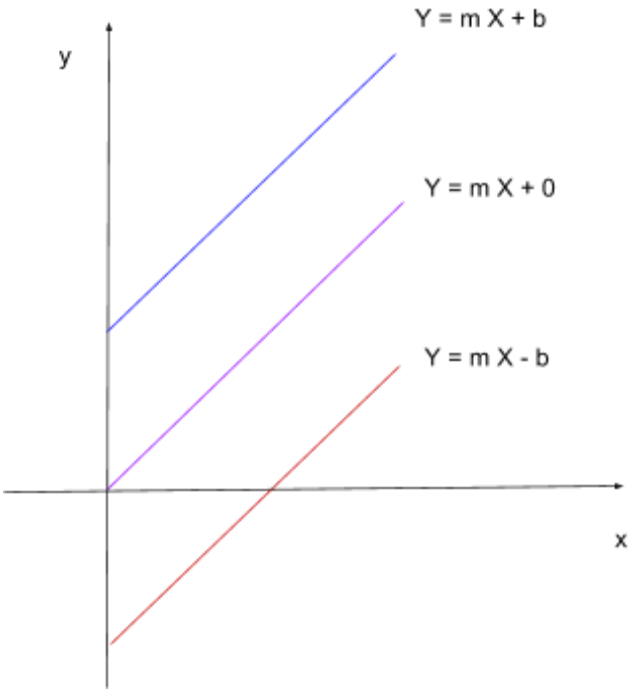


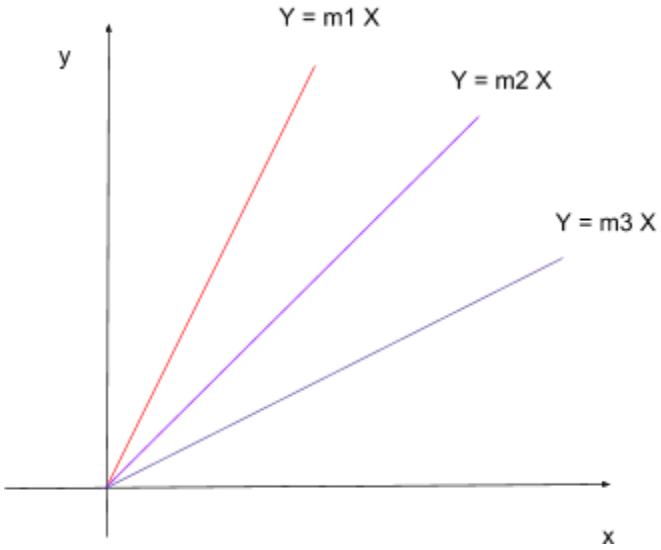
فهي بتديلك قدرة علي انك تتحكم في Output من غير أي اعتماد علي previous input وده حاجة مميزة جدا ..

بس لو حابب تفتكر انت شوفت Bias دي فين قبل كده فانت لو خدت مثال لأبسط شكل Neural Network هيبقي عبارة عن input 1 و output 1 فهيبقي Y= w1 x لماتمثلها graphically هتلاقيها عبارة عن خط مستقيم Linear function

انك تتحكم في W1 هتديلك فرصة انك تتحكم في ميل الخط

يعني في الزاوية يعتبر .... طيب لو عايز اعمل Shift اصلا لل Graph كله ؟ كنا ساعتها بنضيف Bias term عشان المعادلة تبقي Y= mx+b اللي هي هنا y =w1 x +b



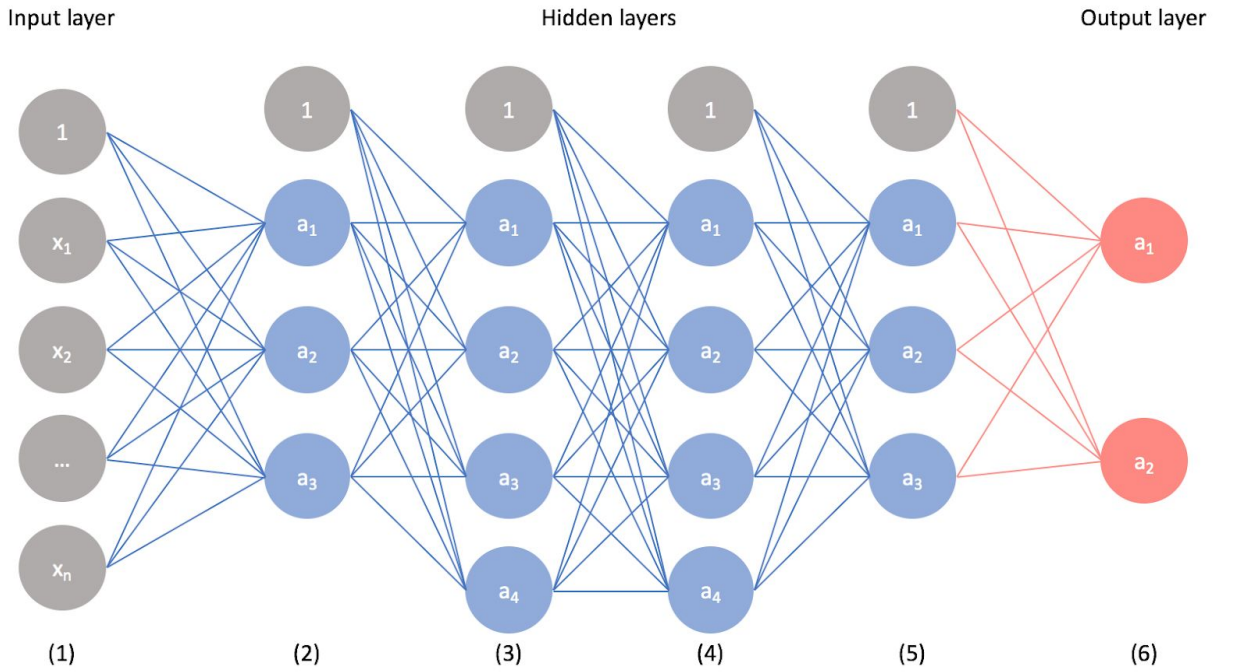


بيقي كده وصلنا ان احنا يعتبر في كل شغلنا بنضيف Node زيادة وهي مش متوصلة بال previous layer وبيبقي ليها weight بنسميه bias وبيبقي ليها input و قيمته 1

بيقي المعادلة النهائية اللي بتمثل Perceptron

Y = w1 x1+ w2 x2 + w3 x3 +1 * b (for Perceptron)

تفتكر احنا بيبقي عندنا perceptron واحدة ؟ لا غالبا بيبقي عندك اكتر من perceptron في Layer الواحدة وكلهم بيبقوا متوصلين ب inputs بتاعتك وبناخد output بتاع كل واحدة ونغذي بيها ال Layer اللي بعدها واكنه input بتاعهم وهكذا



Input layer (1) , Hidden Layers (2:5) , Output Layer(6)

الشكل اللي قدامك ده بيمثل MLP اللي هي Multi Layer Perceptron .....دلوقتي المفروض يبدأ يجيلك سؤال ؟ هو ان في عدد كبير جدا من weights و biases موجود ... طيب هو مين اللي بيحسب قيمها ؟ لان اكيد كل ماهتغير فيهم كل ما هيتيغير Outputs

في الحقيقة مش انت اللي بتحسب قيمهم .. انت بس بتعمل Random initialization ليهم يعني بتفرض أرقام عشوائية

ليهم كلهم و عن طريق Gradient Descent اللي موجود في Back Propagation بيبدأ يتعدلوا للرقم الأقرب للصح ..

لحد دلوقتي احنا مشرحناش ايه هي Back ولا حتي Forward propagation بس بما اننا بنعمل explode لأشهر أساسيات Neural Networks فاحنا لازم نعرف دلوقتي يعني ايه gradient Descent وايه أنواعه وده اللي احنا هنعرفه بشكل أوسع في Optimization .

Optimization

في علم Data Analysis بشكل عام دايما بيبقي عندنا Error وال Error ده بنمثله ب equation وبنسميها Cost function

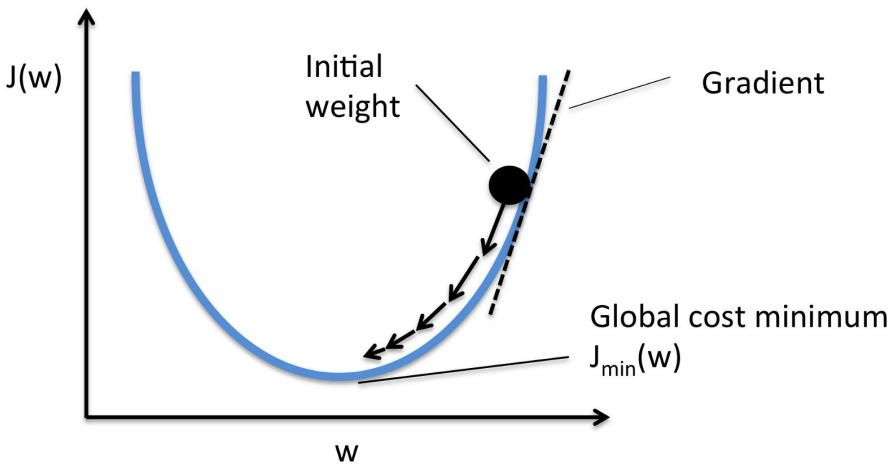
فكلمة Optimization المقصود منها انك تعمل reduction لل Cost Function دي او ساعات بنسميها Losses بس تعالوا نفهم ازاي بنقلل Error ده لو قلنا ان انت دلوقتي مثلا معاك Output معين Predicted وليكن قيمته 60 والمفروض الناتج الصح الحقيقي Actual يعني هو 100

يبقي بالنسبة لك Error لو خدنا احد أشكاله وهو انك تطرح بس ال 2 من بعض فكده

Error = Predicted - Actual

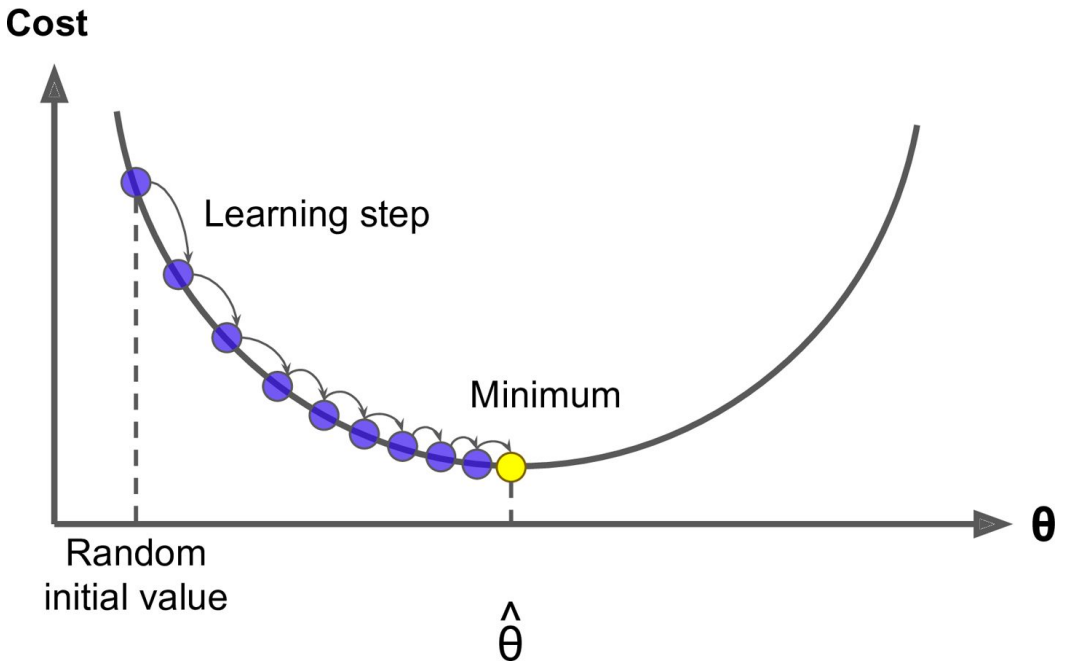
فلو حاولنا نحسب ال error هيطلع معانا 60-100 يعني 40

فتخيل معايا دلوقتي error ده قيمته اتقاطعت مع ال graph في نقطة عند initial weight معين .



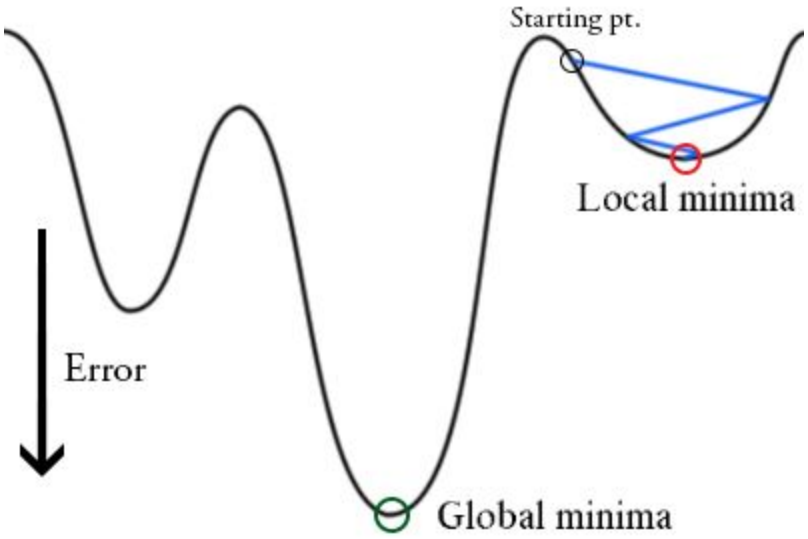
فانت هدفك تخلي النقطة دي تحت عند أقل error اللي هو بنسميه Global Minimum ...
بس هو ايه اللي هيوصلنا لتحت ايه اللي يعرفنا ؟

في الحقيقة Gradient descent هو بالنسبة لك عبارة عن Vector بيوجهك لأقل Error يعني لو بنتكلم عن concave function زي دي من الدرجة التانية فهو لما ناخد derivative بتاعها هيبقي مجرد vector بيشاور لتحت فانت هتبدأ تنزل ب Rate معين بنسميه Learning Rate أو Alpha



بس Learning rate ليه قصة لازم نفهمها

احنا في بعض Costfunctions هنتكلم عن انها مش مجرد Curve لا دي هي مثلا 3d بالشكل ده



فلو قلت ان ال Step بتاعتك هي Learning rate

تخيل لو رقم صغير فانت هتقدر توصل minimum .. بس في الحقيقة ممكن تقع في Local minimum >> اما لو كبيرة فهي ممكن تخليك تعمل jump من local لل global minimum وهو ده المطلوب لانه أقل Error

+لو هي رقم صغير أوي فممكن متلحقش تتعلم يعني فترة التدريب تنتهي وانت لسه موصلتش لتحت ... اما لو رقم كبير فممكن متعرفش تقف بالظبط عند أقل نقطة ...

فالموضوع ليه حلول كتير ومشاكل أكتر هنوضحها فيما بعد

بس تفتكر هل انك تبدأ ب learning rate كبير وتصغر بعد كده ممكن يعالج المشكلة ؟؟

Activation function

فاضل جزء بسيط اوي هنتكلم عنه وهو لو فاكرين في مخ الإنسان قلنا ان كل Neuron بيبقي متوصل بالاف من Neurons جيرانه يعني ويقدر يتحكم في firing بتاعهم .. فده بيديلك فرصة علي التحكم في مين من Neurons يوصل المعلومة ومين لأ ....

بالنسبة لنا احنا هو مفهوم مشابه لل Filters

Step Function

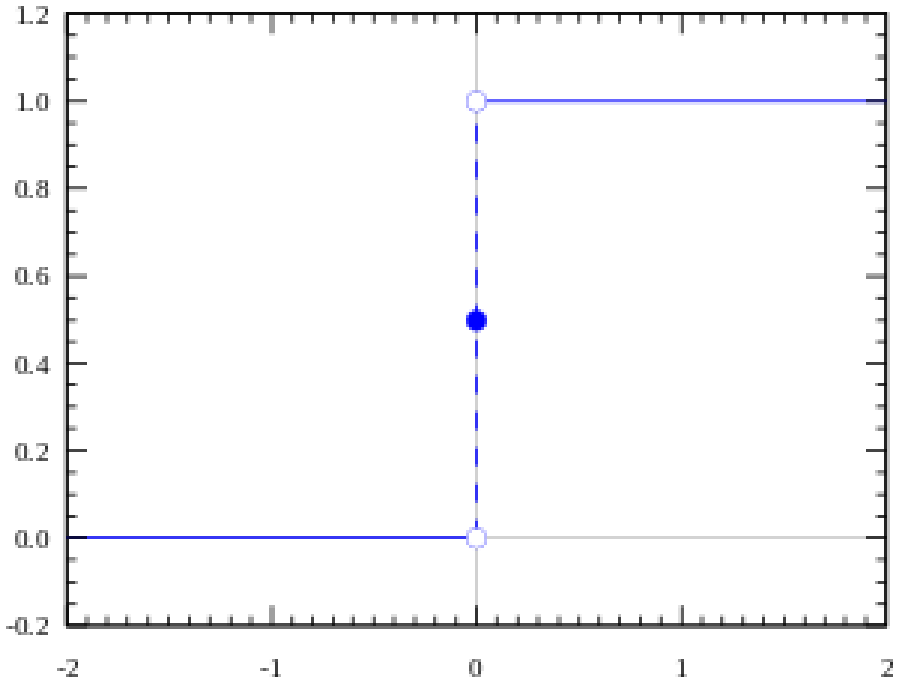
احنا لما فكرنا كده قلنا نعمله زي فكرة Threshold بمعني

لو مثلا عندك درجات اكتر من طالب .. ففي طلاب جايبين 40% وفي طلاب جايبين 70% وهكذا .. فانت عايز تعمل ايه ؟ تحط درجة نجاح Threshold يعني اللي بيقي اعلي منها بيقي ناجح

يعني 1 واللي أقل منها بيقي ساقط يعني 0

يعني في بعض القيم لما تدخل عندك في Activation function هتطلع ب 0 والباقي ب 1

مثال تاني يعني زي الرسمة اللي تحت ان لو input X أكبر من الصفر هيطلع الناتج 1 ولو ال input X اقل من 0 هيطلع الناتج 0 بص كويس علي X axis وعلي Y value



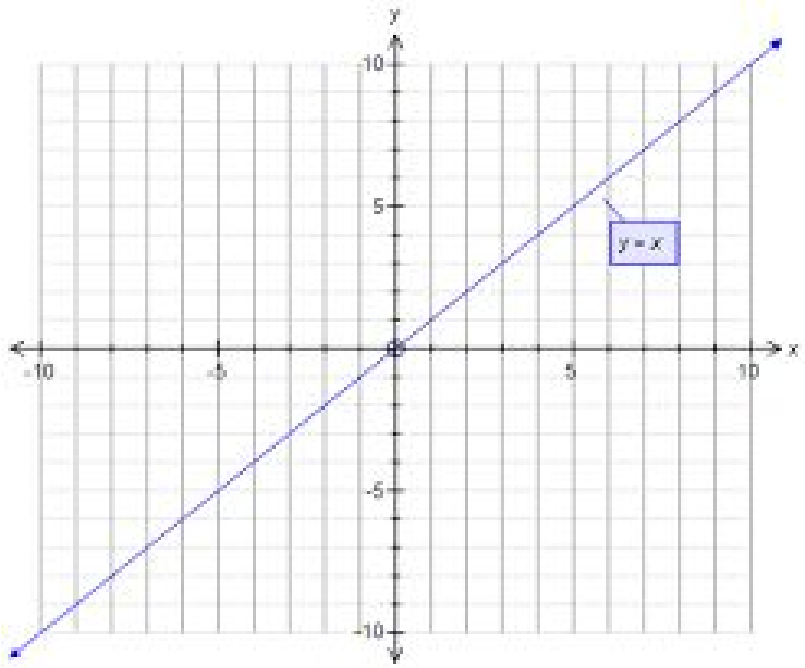
تمام أوي كده انا قدرت اعمل filter وممكن يبقي مثلا علي output layer ويعمل classification مابين حاجتين .. بس عيب

حقيقي لل Step Function هي اني مش بقدر أعمل اكتر من Binary Classification يعني Dog/Cat , Male / Female لكن لو عندي أكتر من 2 زي مثلا خمسة Classes ساعتها هيفشل فبدأوا يفكروا في حل تاني زي Linear Function

Linear function

لان احنا كنا محتاجين Analog output يعني بيختلف معايا فده هيوفرلي احتمالات من -inf لحد + inf

فانت تقدر تقول ان هنا كل قيمة لل x ليها Class مختلف في Y



مثال لو قلنا ان احنا عندنا Classes ليها قيم
Egypt = 0.65 , France 0.9, Germany 0.1

فلو طلع عندك ال input 0.5 ولو افترضنا ان ميل الخط 45 درجة هنقول كده output 0.5 برده يبقي لو قربنا الرقم لأي Class هيطلع عندي الناتج Egypt

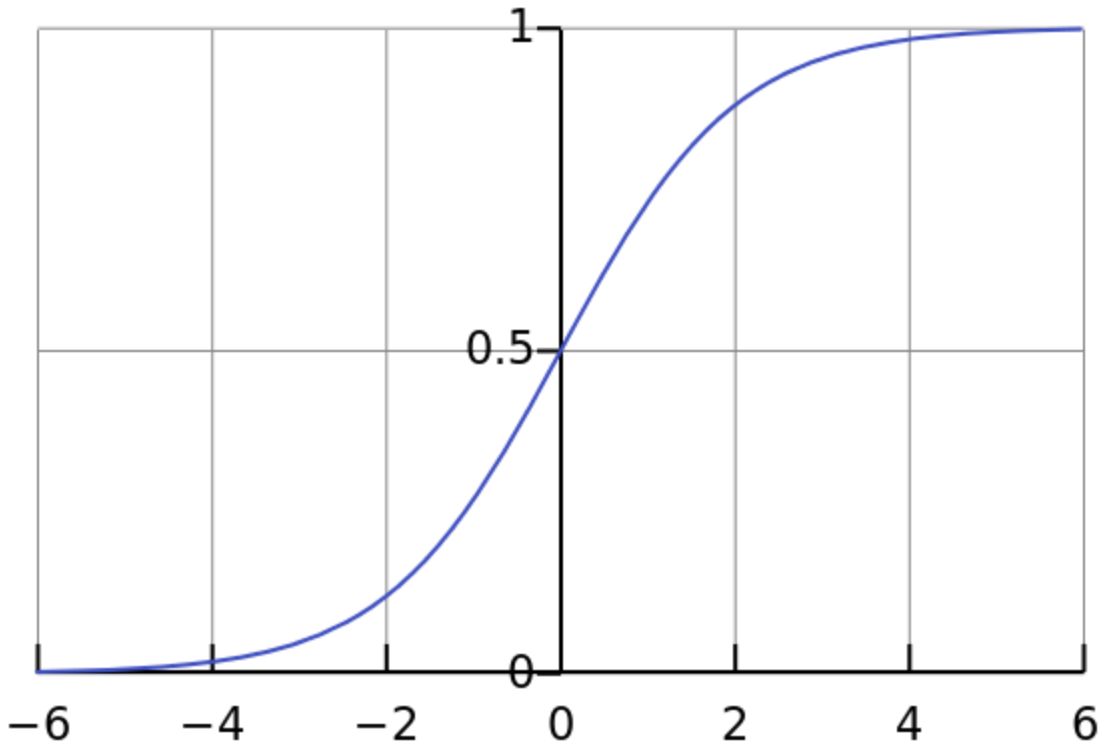
بس ايه العيب ؟

العيب الحقيقي ظهر لما كنا بنتكلم عن Gradient Descent ..
فهو في الحقيقة هي عبارة عن Derivative لل Activation Function فطبعا لو عملت اشتقاق للخط المستقيم الناتج هيبقي ايه ؟ Constant >> وبالتالي مفيش حاجة هتعرفني اتجاه اقل error فين ...

فبدأنا نفكر في نوع تاني يحل كل المشاكل اللي فاتت دي وهو

Sigmoid function

فهي دي بالنسبة لنا الحل نشوف شكلها كده



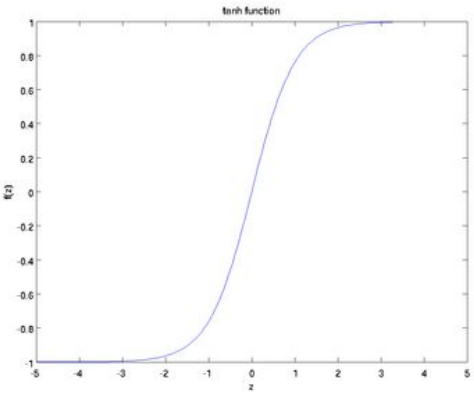
بتوفرلنا موضوع threhold في انها بتنتهي ب 0 و 1 يعني ممكن اعمل filter .. كمان بتوفرلي Variance range هتلاقيه مايل شوية وده بالنسبة لي Analog يعني كده تمام اقدر اعمل اكتر من Classification ... واخيرا بقي هي non linear فحتي لو عملت ليها derivative هتبقي vector وهو ده اللي انا عايزه ..

بس هي برده محكومة ب Range ان اخرها 0 ل 1 طيب افرض انا عايز حاجة بالسالب ؟

فظهر نوع تاني وهو

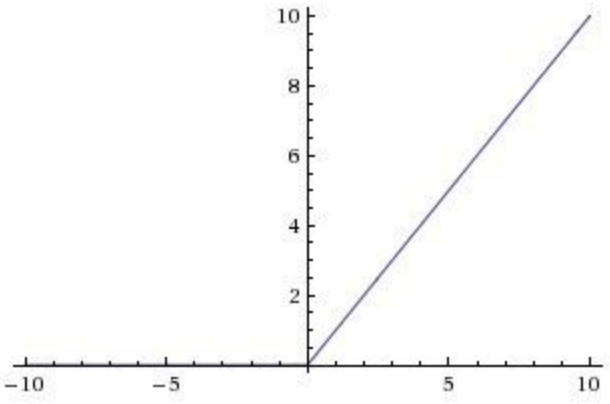
Tanh function

وهي هتوفرلنا Range أوسع من Sigmoid بس ميل هيختلف شوية .. ليها تطبيقات كتير بس مش هنحتاج نخش في تفاصيلها دلوقتي



Relu Function

وهي اخر نوع حابيين نتكلم عنه دلوقتي بياخد ميزة Range ال Linear function وفي نفس الوقت ميزة Filter ودمجهم مع بعض ب Range from Zero to - inf



بالشكل ده طبعا لو فكرت تاخد اشتقاق الخط الثابت اللي عند zero ده فانت بتعمل اشتقاق ل Constant وطبعا الناتج هيبقي صفر ...

ودي بتوجهنا لمشكلة بنسميها Dying Relu ... هنتكلم عنها فيما بعد بشكل اوضح بس Relu function من أشهر Activation functions المستخدمة

دلوقتي احنا بقينا جاهزين نفهم ايه PRocess اللي بتحصل عشان Neural Network تتعلم وتوصل ل Solution

وده بيحصل علي مرحلتين

* Forward Propagation
* Backward Propagation

Forward Propagation

وهي بالنسبة لنا كل الخطوات اللي بتتم من أول مابنعمل Random initalization لكل Wieghts لحد مانوصل لل Output مرورا بكل Perceptrons بتاعتنا وال Activation functions والقصة دي كلها

بمعني ان :-

هدف Forward Propagation هو الوصول ل Output بعد سلسلة من العمليات ..

بس هل Output ده صح ؟ يعني هيساوي Actual ؟
أكيد لأ لانه مبني علي Random weights .....

Backward Propagation

وهي الهدف منها انك تعمل update لل Weights اللي عندك من الاخر للأول يعني بهدف تقليل ال Error وده هيتم أكيد وسط Algorithms من Gradient Descent و parameters تانية زي Learning Rate

هل بعد ماعملنا Backward propagation كده احنا خلاص يعني جاهزين لو عملنا Forward مرة تانية ان احنا نوصل للهدف بالظبط ؟

أكيد لأ برده انت بس ممكن تبقي قللت ال error فهتحتاج تكرر الموضوع ده مرة و اتنين وتلاتة وهكذا

اللفة المكونة من Forward Propagation + Backward Propagation احنا بنسميها Epoch

كل مابتزود عدد Epochs بيفرق كتير معاك في الوقت والجهد علي Processor ده غير انك برده ممكن تتجه ناحية مشكلة كبيرة أوي وهي Over fitting .

لحد هنا ممكن نقول احنا نكتفي بالمعلومات دي بشكل مبدأي في علم Neural Networks وممكن نبدأ نتجه ناحية Coding شوية ونشوف هنقدر نوصل لايه باذن الله

Develop Your First Model With Keras

باذن الله هنشتغل Python Programming Language وده نظرا
لأسباب كتيرة جدا منها سهولتها و كثرة Libraries بتاعتها
وانها حصلت علي لقب لغة Deep Learning وده طبعا نظرا
لانها Open Source

و طبعا لكثرة المكتبات
وصعوبة تنزيل بعضها فاحنا
هنتجه باذن الله ل Platform

اسمها Anaconda

هتوفرلنا IDE زي Jupyter
وكمان هتدينا Library للtool
installation وهي Conda
وكمان هتساعدنا نبني أكتر
من Environment بشكل بسيط جدا جدا

Some Installation Tips:-

أول خطوة هي انك تنزل Anaconda سواء انت Linux أو Windows أو Mac

https://www.anaconda.com/download/

بعد ماهتخلص installation لو انت Windows هتتابع معانا من Anaconda Prompt
أما لو Linux فهتابع معانا من Terminal عادي

لو عايز تتأكد ان كل حاجة نزلت صح أكتب Python في Terminal/Anaconda Prompt

لو ظهرلك

python3.6/ Anaconda-Custom ... etc

اعرف ان انت كده تمام وجاهز انك تبدأ

هنشتغل باذن الله من Jupyter lab

هتختار python3 من Notebook وكده انت جاهز رسميا نبدأ سوا أول برنامج لينا

Your first Neural Network with Keras (Diabetes Classifier)

-Most of Code are taken from here-

هنبدأ دلوقتي نحاول نعمل كود بيصنف هل العينة اللي قدامه دي مصابة بمرض السكري ولا لأ

ايه الخطوات اللي هنمشي عليها ؟

1- Load Data

2- Data preprocessing

3- Build the Model

4- Training

5- Prediction

الكود ده مجرد عرض لشكل Neural Network مش الهدف منه انك تتعلم Coding وتبقي جاهز هو مجرد عرض فقط وتاخد فكرة بسيطة عن شكل الكود و Parameters بتاعته

1- Load Data

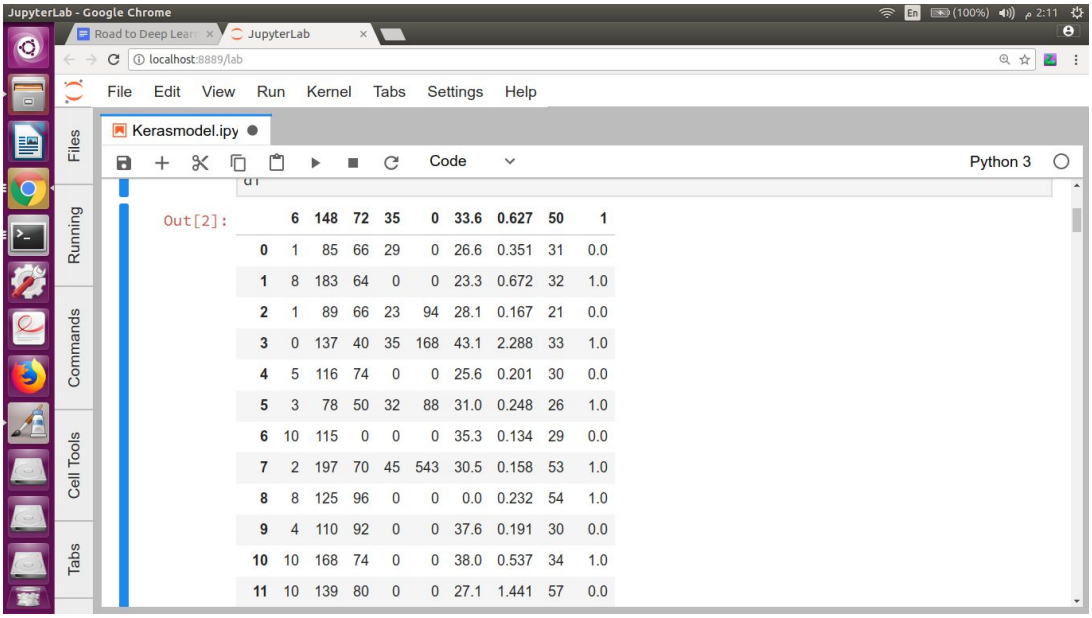
طبعا عشان تعمل Load لل Data الموضوع بيبقي متوقف علي العميل بتاعك موفرلك Data صيغتها ايه احنا في حالتنا دي متوفرلنا csv file

وهو diabetes.csv .. حمل Data من هنا < Dataset

أبسط طريقة انك تقرا File ده هي انك تستخدم function اسمها read_csv بس هي مش متوفرة في Python فهتضطر تناديها من library اسمها pandas بالشكل ده

```
import pandas as pd
df = pd.read_csv('diabetes.csv')
df
```

تعالوا نشوف الناتج عامل ازاي ... في الحقيقة قدامنا Table محترم متقسم ل 9 عواميد بالشكل ده .. بس في مشكلة ان Data نفسها جايالنا من غير header فاحنا هنضطر نجيب headers نفسها عشان الدنيا تترتب شوية



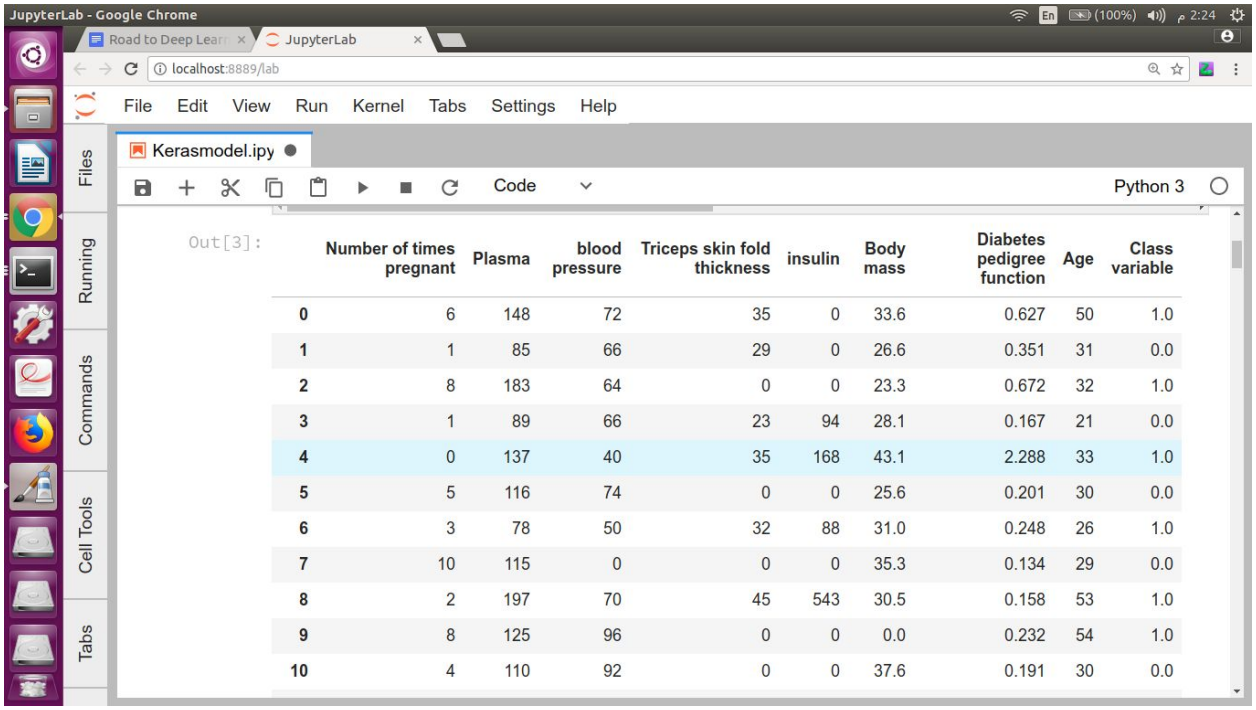
	6	148	72	35	0	33.6	0.627	50	1
0	1	85	66	29	0	26.6	0.351	31	0.0
1	8	183	64	0	0	23.3	0.672	32	1.0
2	1	89	66	23	94	28.1	0.167	21	0.0
3	0	137	40	35	168	43.1	2.288	33	1.0
4	5	116	74	0	0	25.6	0.201	30	0.0
5	3	78	50	32	88	31.0	0.248	26	1.0
6	10	115	0	0	0	35.3	0.134	29	0.0
7	2	197	70	45	543	30.5	0.158	53	1.0
8	8	125	96	0	0	0.0	0.232	54	1.0
9	4	110	92	0	0	37.6	0.191	30	0.0
10	10	168	74	0	0	38.0	0.537	34	1.0
11	10	139	80	0	0	27.1	1.441	57	0.0

وده عن طريق Data Set Details من اللينك ده

[Data set Details]

فهنعدل في read csv function وهنخليها بالمنظر ده

```
df = pd.read_csv('diabetes.csv', sep=',',
                 names = ["Number of times
pregnant", "Plasma", "blood pressure",
"Triceps skin fold thickness",'insulin','Body
mass','Diabetes pedigree
function','Age','Class variable'])
df
```



	Number of times pregnant	Plasma	blood pressure	Triceps skin fold thickness	insulin	Body mass	Diabetes pedigree function	Age	Class variable
0	6	148	72	35	0	33.6	0.627	50	1.0
1	1	85	66	29	0	26.6	0.351	31	0.0
2	8	183	64	0	0	23.3	0.672	32	1.0
3	1	89	66	23	94	28.1	0.167	21	0.0
4	0	137	40	35	168	43.1	2.288	33	1.0
5	5	116	74	0	0	25.6	0.201	30	0.0
6	3	78	50	32	88	31.0	0.248	26	1.0
7	10	115	0	0	0	35.3	0.134	29	0.0
8	2	197	70	45	543	30.5	0.158	53	1.0
9	8	125	96	0	0	0.0	0.232	54	1.0
10	4	110	92	0	0	37.6	0.191	30	0.0

2-Data PreProcessing

لحدهنا احنا خلصنا خطوة Loading Data وكمان يعتبر بدأنا في Pre Processing بخطوة تعديل ال Header

الخطوة اللي جاية دلوقتي ان احنا هنحاول نقسم جزء Features لوحده وهو X وجزء Y لوحده وهو Label

في function عندنا اسمها iloc هتحتاج منك object اللي هو table /Matrix بتاعتك يعني عشان تبدأ تتحرك فيها وتاخد جزء من Matrix وتفصله لوحده لو عايز

فانت دلوقتي عايز تاخد علي سبيل المثال X

هي أول 8 أعمدة و مثلا اول 760 صف

فهتدخله

From 0 : 760 in rows and 0: 8 in columns

اما Y فهدخل

From 0:760 in rows and the 8th column

```
x=df.iloc[0:760,0:8]
y=df.iloc[0:760,8]
```

تقدر تعمل Check بانك تشوف قيمة X او y ايه هي !

3-Build the Model

فهو في الحقيقة احنا نفترض عايزين نعمل Model مكون من

Input Later - 8 Input Neurons

First hidden Layer -12 Neurons (Fully connected / Dense) with relu

Second hidden layer - 8 Neurons (Fully connected/Dense) with relu

Output layer - 1 Neuron (to tell if its 0 / 1) with Sigmoid

في الحقيقة احنا هنستخدم Keras وهو بيتبع أسلوب بنسميه Sequential نقدر نقول ان احنا بنعمل Sequential Model وبنبدأ بعديه نضيف Layer ورا التانية

```
model = Sequential()
model.add(Dense(12, input_dim=8,
init='uniform', activation='relu'))
model.add(Dense(8, init='uniform',
activation='relu'))
model.add(Dense(1, init='uniform',
activation='sigmoid'))
```

فاحنا ده بالظبط اللي قلنا فوق ماعدا كلمة uniform ممكن نعديها دلوقتي ونرجعلها في Week 2

لكن بالفعل احنا عملنا أول Layer مكونة من 12 نيورون وهي منتظرة من input كام ؟ منتظرة 8 ونوع Activation هو relu وهكذا باقي السطور وطبعا زي ماقلنا بنعمل Sequential model وبنبدأ نضيف Layer ورا التانية ...

3- Train the Model

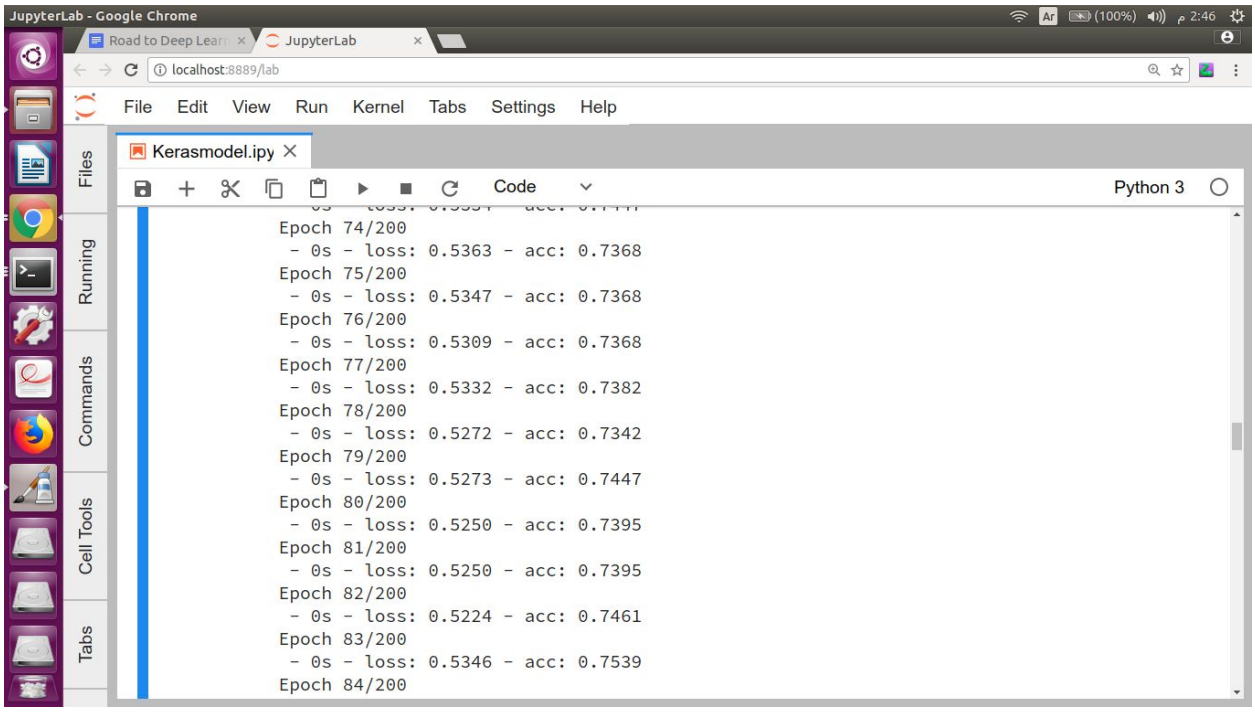
دلوقتي هنتنقل لخطوة Training احنا مش interested دلوقتي في كل تفاصيلها اكتر من اني بجهز فيها Optimizer بتاعي سواء gradient descent أو غيره وكمان بدخله cost function بتاعتي سواء هي طريقة حساب error العادية او شكل تاني زي binary cross entropy وهنتكلم عنها فيما بعد

وبدخل فيها موضوع Epoch واشوف عايز كام لفة و اعرف ال model شكل X ,Y اللي هو features , labels بتاعتي

```
model.compile(loss='binary_crossentropy',
optimizer='adam', metrics=['accuracy'])

model.fit(x, y, epochs=200, batch_size=32,
verbose=2)
```

وهيبدأ بالفعل يعمل Training بالشكل اللي تحت ده



كده فاضلك خطوة اخيرة وهي انك لو حابب تعمل Prediction

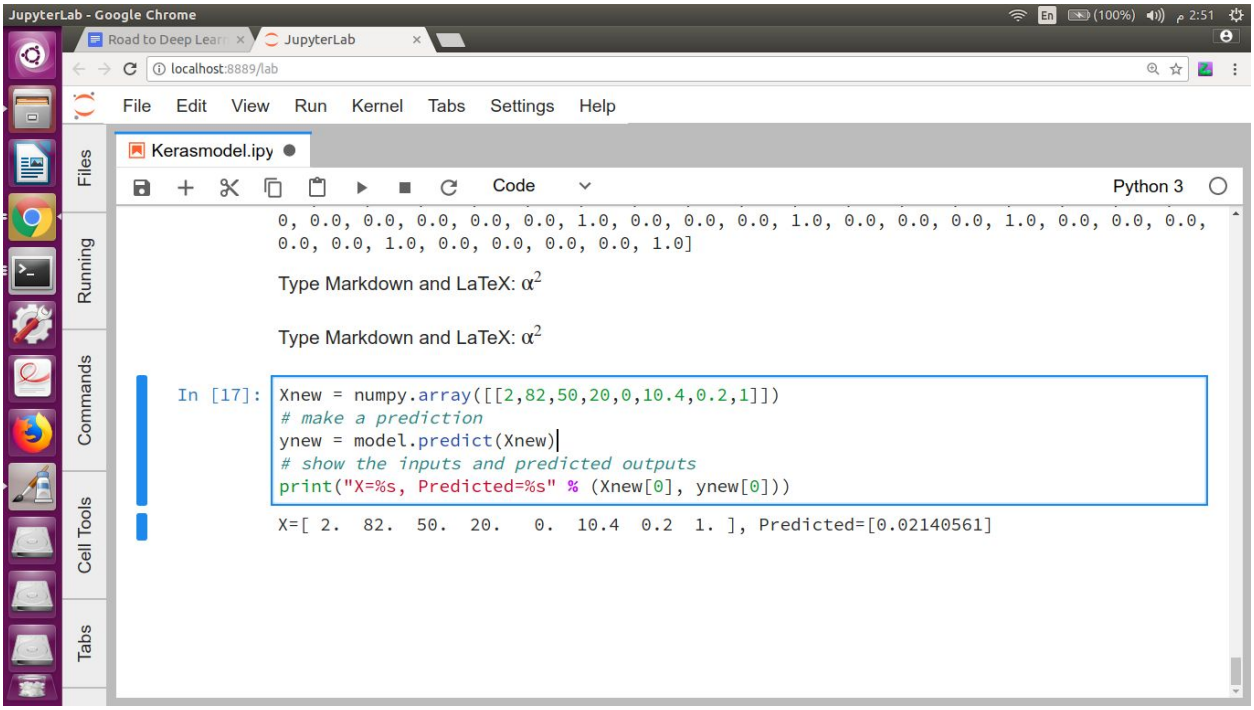
4-Prediction

فانت دلوقتي ممكن تاخد أي سطر من Data وابدأ غير في أرقامه ودخلهاله في array وشوف هيعمل prediction بكام طبعا لو اقرب لل 0 يبقي هي مريضة لو أقرب لل 1 يبقي هي سليمة

```
Xnew =
numpy.array([[2,82,50,20,0,10.4,0.2,1]])
# make a prediction
ynew = model.predict(Xnew)
# show the inputs and predicted outputs
print("X=%s, Predicted=%s" % (Xnew[0],
ynew[0]))
```

عدل في ال أرقام بتاعت array براحتك وشوف كل مرة هيطلعلك prediction عامل ازاي

وتقدر تضيف function round للتقريب يعني



لو في أجزاء من الكود مش واضحة ده طبيعي الهدف النهائي كان انك بس تشوف الكود بيبقي ماشي ازاي وايه علاقته باللي شرحناه بشكل عام ...

هنتعلم في Week 2 أو Chapter 2 :-

- ايه هي Deep Neural Networks ؟
- هنتعلم Convolution Neural Networks !
- هنعمل CNN Code on MNIST Data set .
- هنتعلم Tensor Flow .
- و Concepts كتيرة جدا في CNN .
- و اخيرا هنعمل code hardware with Arduino.

END OF WEEK 1