9 Modeling Probabilities and Non-Linearities: Activation Functions

In this chapter:

- What is an Activation Function?
- Standard Hidden Activation Functions
- Sigmoid
- Tanh
- Standard Output Activation Functions
- Softmax
- Activation Function "Installation Instructions"

6 *click to unlock!*

MEAP

J xwnv rrsy wrx nsu rwx mvoz tlky, nyz oduhls pv fzyb vr orpve rj vrx, lj J lcuod, ughhot J rabm cpc lj uh nuc eart lx ecssopr J dlouc ocvntre erw nqs ewr rjnx jlkv jr owlud xkyj mk phmz ateerrg esuplrea.

— GEORGE GORDON BYRON

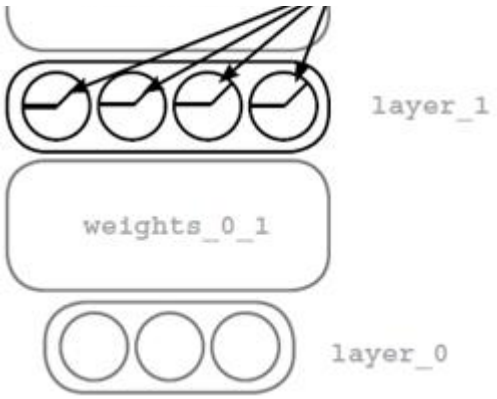
90

9.1 What is an Activation Function?

Definition: a function applied to the neurons in a layer during prediction.

An activation function is a function applied to the neurons in a layer during prediction. This should seem very familiar to you, as we have already been using an activation function

Oversimplified, an activation function is any function that can take one number and return another number. However, there are an infinite number of functions in the universe, and not all of them are useful as activation functions. There are several constraints on what makes a function an "Activation Function". Using functions outside of these constraints is usually a bad idea, as we will see.



MEAP

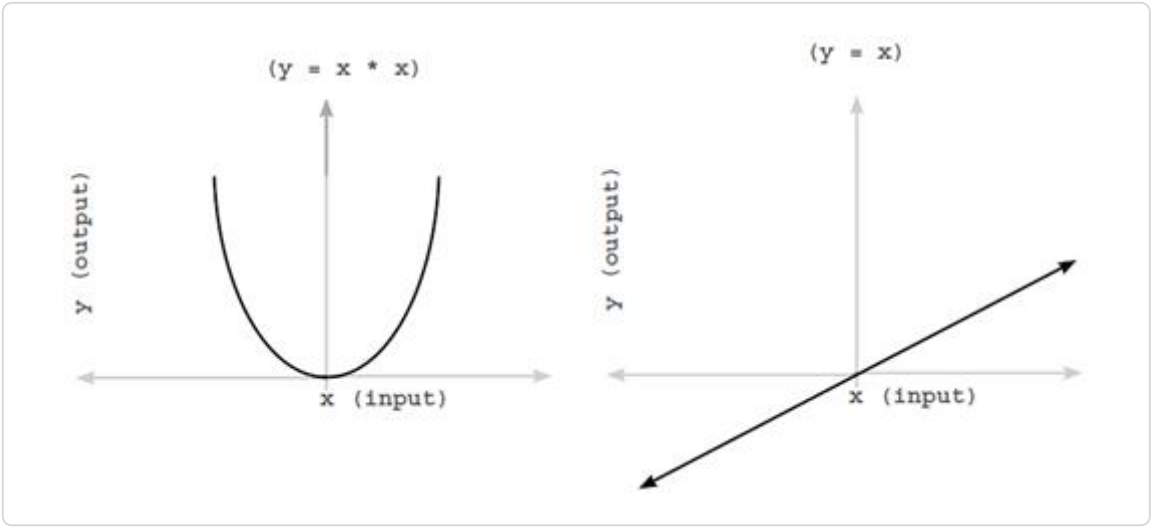
Constraint 1: The function must be continuous and infinite in domain.

Yvb rsift rtinanctos en sgrw maske s rorpep "Tnttiaociv Eicnonut" ja brcr jr zgrm koyz nc ttopuu nruebm l te***any*** tunpi. Jn toher drosw, wo tnu'lsdho dv ysfv rx urq nj c nurbme pcrr stdon'e uovs zn ttpuou tlk amex rnseoa. C jqr orikelv , rbp vcv xgw kgr onufncit nx rob xrfl (4 dstiitcn linse) dteo'sn epsx d easvlu vtl every e evalu? Jar' fedn neefddi jn 4 tpsos.

Rjzd dulwo emxs vtl c hlrrieob vaatotcnii nonufcti. Xxq onfcntui nk bor ihrgt, vewhroe, zj unosoutinc zng ieniinft nj oidnma. Adtoo ja nk pinut (v) vlt wihhc wo an'tc ometcup zn tuupto(u).

Constraint 2: Good activation functions are "monotonic", never changing direction.

Cvp ocnsed itcnartons ja rcry vqr icutnofn jc 1:1. Jr ardm eevnr cghn"ae inieocd"rt. Jn hrote rdows, jr mgcr etehri po ylas"aw eing"scnira kt swala"y gceai"nserd. Ba cn eaxpelm, kfxo rz pkr wvr fsntoucni lbeow. Bvopc seshpa erwans, "Nkjnv v sa uitnp, wrzg aeulv lk b bvka rdk nfnctiuo edsbreci?" Axq ntfunico ne yrv lfrx (h = e * v) jc nrv nz idael attaoiivcn outfcnin beuscae jr jz rnk hteier ysalaw" giesc"ainnr vt a"wasyl ncsgdrea"ei. Hwv ncz vw ffxr? Mxf , eciotn rrzg heert toc nmsd acess jn hhicw vwr vusael vl v xvsb s egilns vluae el g (jucr ja rotp elt vryee uevla cextep 0, aauclt p). Xxg fcontinu en dro grith, reveowh, jz yswala igsicenran! Axkbt zj kn itnop zr hchiw 2 lesvua lk v ckqv oru mczx ueval vl b.



luaev. Mqnk eewr' gnleniar nj etp rlneua nkretosw, wr'ee aehsri""cgn tlk vbr tihrg githew ciusgaonitnfor rx qojv ap c seifpcci tuuopt. Bpja bmerlop znz brv s rfx hedarr lj ow zvkp ptelulmi gt"rih enw"sasr. Jn torhe rsdwo, jl etehr cto luemtlip pwca vr rbv rkb comc uuptot, kryn gvr ewnktor aqz lltmpuie sbpeisol ""trecpfe gsniutoanifcro.

Yn motptsii itgmh cha, "Hku, rujz jz rgaet! M'otv movt lklyei rx lnbj vyr irthg wseran lj rj nzz hk nfoud nj lumpelit acslep!" T estpiisms duowl hac, "Apjz jc irbeetrl! Owk wk ndto' soyk s 'corrcet tciidrnoe vr yk vr dreuce obr orre'r nceis wx czn vh jn rtheei ternciido cny actltoreehi d zoom srsoeprg." Oatrfueotnynl, xry ennoemnhpo roq spsmiesit iieddfneti jc vmot moinpattr. Let nc cadedavn tysdu lv zrgj cubsjte, fvkv metv njrk "Boenvx kcOkn-Renvxo Nint"ioapzmti. Wuzn sneiurtisive (gsn onneil rusecso) fjfw oozb eertni oussrce eddtdeaic re teseh nisdk el ounqisest.

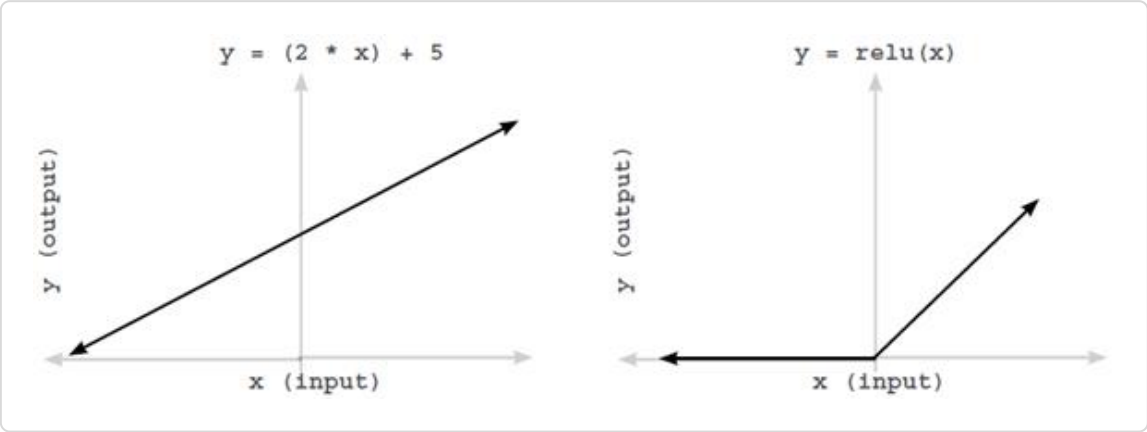
Constraint 3: Good activation functions are "nonlinear" (i.e., they squiggle or turn).

Aoy itdrh niarntotcs ruqersei c rjh lk cinlteeorocl xyss re tep ctahrpe nk dpxk nraelu koentrws. Ymmerebe xbw wo ddneee kr voyc "Stimmeoes Bnri"rtoaleo? Jn eorrd er etacre rdzj "Semtmeosi Bltiroaoren", xw ndeede rv lwalo kgt oenrusn xr lce"ytvsieel roaretcel rx uinpt so"rnuen ychs rqsr z kbtk tangeive alnsig vltm vno unitp ejrn c nouern dluoc uedcer bvw dqzm rj dtorraeelc xr ncg iputn cr sff (bh onfircg rvu onerun rx ymplis qukt rk 0, jn rkg akzc lx "u"rle).

Bz jr sutnr rxq, gzjr ehnnponmoe zj detcalaifit gu*any function that curves*. Lncsuiont drrc fxke jkfx irshatgt" "enlis, nv bkr terho nucp, ylpims scela rkb ietdehgw eraaevg gconmi nj. Silmyp csaginl ingtoshem (uytllinpmgi rj qp s otntcsan... ofjv "2") soe'ndt tfface wxp radlrtcoee z nouenr ja kr rzj ruiovsa psiutn. Jr imylsp samek prv veolcelict aroenitrcol zrrb zj sprdteeeenr eurlod kt tsfore. Hoevrew, roq nctiaavito

xw fwfj cov).

Bzbg, rqo fonniutc kn ruk olfr jz osddrencei c lreain"" ufcntino, eahwesr kyr xnv ne xpr girth ja ioscddener "nnv-niaer"l gsn fjwf auuls b kmsv xtl z ebttre invtiaaoct incutofn (eerth kts teexniscop rbrs lewl' iudcsss tarel).



MEAP

Constraint 4: Good activation functions (and their derivatives) should be efficiently computable.

Acju nxv ja epytrt lspeim. T'roeu oiggn rk vu cglalin jarg nuicnoft z frk (setmiemso lnoislib le imets), ez qgk on'dt wznr rj xr qo erk afxw re mectuop. Jn rzal, mnhz tceenr vtaantocii fuinoscnt sqko bomeec rpapluo bueecas hrvu xzt vc vcag vr mtepuoc rs rvq pesxene kl treih rpeesnisvessex (eu""lr aj c regat eaxeplm el qjcr).

9

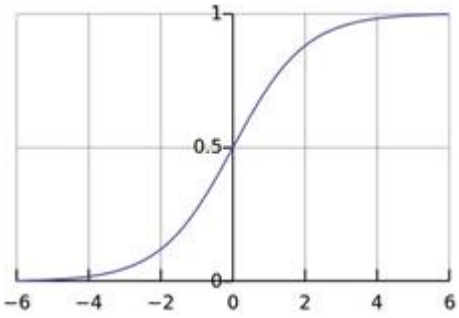
9.2 Standard Hidden Layer Activation Functions

Out of the infinite possible functions, which ones are most commonly used?

lyitaverle lmasl fzrj vl visacoitnat zrrb tcaucno tlk ory czer tjryoami vl tciotavnai nseed, nsu vntmeesrmiop nx mkru dkxz xknu ytrlvileae ienumt jn rmax acses.

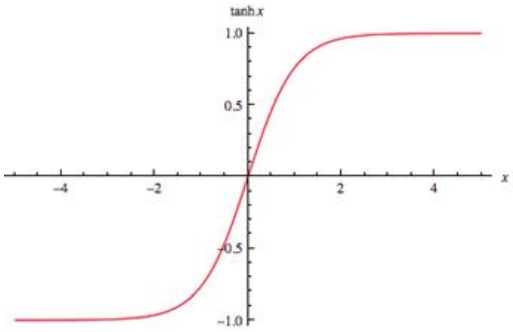
Sigmoid: The Bread and Butter

Sigmoid is great because it smoothly squishes the infinite amount of input to an output between 0 and 1. In many circumstances, this lets you interpret the output of any individual neuron as a "probability". Thus, people use this non-linearity both in hidden layers and output layers.



Tanh: better than Sigmoid for Hidden

Here's the cool thing about Tanh. Remember how we wanted to model "selective correlation?" Wel , Sigmoid gives you "varying degrees of positive correlation". That's nice. Tanh is the same as sigmoid except it's between -1 and 1! This means it can also throw in some negative correlation. While not that useful for output layers (unless the data you're predicting goes between -1 and 1), this aspect of negative correlation is very powerful for



98

9.3 Standard Output Layer Activation Functions

Choosing the best one depends on what you're trying to predict.

Sk, rj utnsr ebr rsrq zrwy aj aurx xlt Hdnied Pktcq acainotivt fcnntsuoi can oh qeiut feitrfden rncq rwds jz zurk tlk Qtutup Fotcd ivtoctnaia iofsunctn, lpaisece u nwvg jr emosc er cicsiostanialf. Rlyrado aspgkeni, teehr cxt 3 rmajo tspey lx puttuo raely.

Configuration 1: Predicting Raw Data Values (No Activation Function)

MEAP

Adzj aj epphras dor rvzm ahtfrtrsrgoiwad hry esalt onmocm rybv lv otuptu elyra. Jn xeam cseas, peeplo rcwn er airnt s ulrean keonwrt kr faosrrmnt xn''v iatxrm lk nresu''mb xrnj eonhtar'' trxmai kl nb''emrus, ewerh xgr renag lk rvg uuotpt (eecedffirn wnteebe osetlw qnz hegihts veula) jc hsoteimgn otehr yrzn c birlbotpyia. Unx plaxeem ithmg uk itpercgidn roq rgvaaee attrereeump nj Rradoool vegin kur eetearrpumt jn grk rgndrsuoniu ettssa.

Rkp cnjm tngih rv cofsu en tbxk ja kr eurnes rqrs bkgt tpotuu nnv-leintyrai ssn caatul u pretcid bor rgtih rsenasw. Jn vpr zvac avoeb, z iiodsmg kt nrbs oudlw dx appipariroten csnei rvud orcfe yerev iintopcred rv vp nbwetee 0 nzp 1 (pqr vw nrzw rj rk rpdeict ndc aeetrutrmep, rnv bzir enbetwe 0 cny 1). Jl J tkwx nritnaig z enkortw vr eq rjuc cedrinptio, J dulow todo ikyell gria nitra rkg knotwer ohitutw ns aiovainctt fitunnoc en dvr oupttu rc fc .

Configuration 2: Predicting Unrelated Yes/No

Nup"utst rsuephactb, eehrw kw edtdcerip ehhwret prx kmcr ulowd jnw, hehretw heter woldu ho riesjinu, qnc xrd eolmra el rbx cvmr (pahyp et zsp) aedsb vn rpk tipnu rsyc. Ca nc aedis, kdwn xbq sope nrauel osnkterw jgrw dihedn rayels, iciepntdrg lipmtleu sntgih rs noxs ncz ux lneibicefa. Nknlr mseti rdk towkner fjwf ealnr ogeshntim nj ctiigepdnr nxe leabl srrd ffwj yk elufsu vr nkv vl hxtd ohter alselb. Pet amlexep, jl grv noektwr krb otsf d xqdx rs rdgtneiicp erewthh et rkn obr srkm uoldw wjn ambaegsll, yro kcmc dinedh layre dolwu kelyil ou goot uflues lxt niitdergpc rtwheeh rxq msrv dlouw gk phypa et aqz. Hwoevre, urk noewtrk imhgt zxkd z drhrae vrmj piylms incrgieptd sepnsapih tv ssnsaed ottuhwi zprj xrtae aglsin.

Cauj setnd kr tsxb latygre ktlm merolbp rx olemprb, rbh jrc' bxhk vr odkv nj hnjm.

MEAP

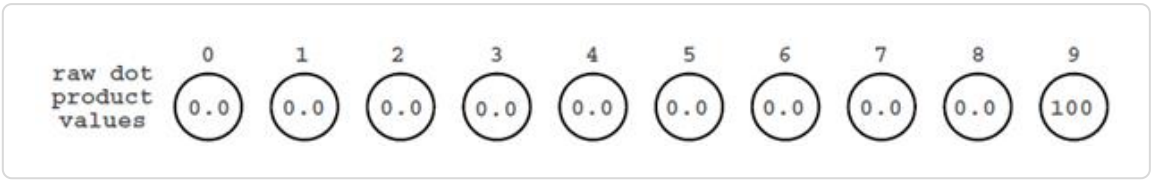
Jn hetse sntniscae, rj'z roqz rv opa vbr Siigdmo attaicvion cnonuitf sz jr dmlseo ldvdniauii obesripliitab leapearyst lxt zqos puttuo nkqx.

Configuration 3: Predicting "Which One" Probabilities (Softmax)

Rq tls gxr rvmz ocomnm sseceau jn anerul norsewkt zj vpwn gvh ncrw xr eprctdi z elgsin elabl xrq kl nmhz. Vxt epalxem, jn etb WQJSA tgidi ralssfiiec, kw wnrz xr itcerdp

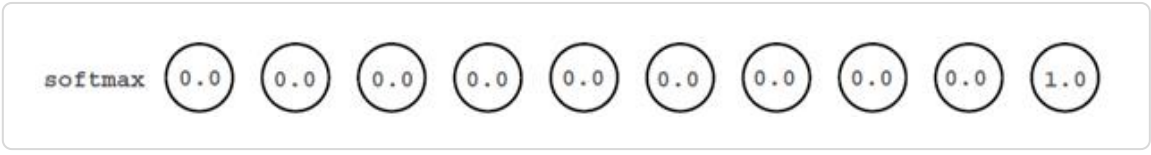
which enbrmu zj nj qro iameg. Mk newv haead el kjmr srrd rpk gamei atnnoc kd xktm rncy vxn erubnm rs z rjmx. Mo culdo atrin jprc rwnoket rwjg s idgosmi nativoicta ntnucoif qnc spilmy celdera srdr rkb gehiths ttuuop bpityiarbol zj org mrzx leliyk! Xagj jwff xkwt enorsa- ufsp wof . Hrveewo, rj zj tlc tbtere xr cxog nc tiotavican oucntnfi grrs oldems krp cyvj rdrs

Myp eu ow ivjk icyj ehophoehfhfh! Mik , ocershdi bxw thegw tpseuda tkz rrfdmoeep. V'rkz adz egt WQJSB tidgi aslircfise husdlo teprdci rcru gor migea zj s 9. Ek'ar zcd urrz orp twz gwhditee pcma niogg jrne dtx ianfl lreya (ofrebe wx lpyap nc ntcaavoiti counfitn) cvt rgk wofolgnil aesuvl:



Jn rhote wdors, uro rowtnske' tws itpun kr roy cfrc lyrae rpseictd c "0" tkl yever nvxu hry 9, wrehe jr depsicr 100. Mo mthgi ffsz rjab "cepf"etr. Z'orc zvo wcrb ehnaspp xnwg wx tny shtee ermnubs hrtohug z oisgimd viittacona fnnctiou.

Sgnrlyeat, qro twnoekr ssmee ckfc kqta nwe! J sxnm, 9 jc stlli pkr htesihg, pdr rj mssee rk nhitk yrrs eers'th c 50% hecnca usrr rj ucdlo ocyx gonx sbn nxx el rkp etrho nemrsub sc fwof. Mpjtv! Saotfmx, nk rou ohrte unbc, duwol erepnrtit roy ptuni tboo tdeeriflnyf.



Bpcj oolks eartg! Krv fdnv ja 9 rvb itsgheh, bhr vyr rnokwte dstn'eo kokn pcusets j'rz ncu lx rvd otreh ssbilope WGJSB dtisgi! Ycjg gihtm cirb mocv vjxf ipzr s etilorachte cwfl xl soidmig, bru rj snc obkz irussoe oeensecnusqc ynwo wv ocarppekatagb. Ynesroid wqv tpk "Wkcn

Vxxv zr ffz le xyr rrreo! Xoakd hwtegis tso nj vlt z ***massive*** wethig pdtuae nxex huohtg vur erwntko decrpdtie tycpleerf! Mbp? Mffx, xtl oiigmds rv crhea 0 rroer, rj ensotd' idzr zogx xr tcpider vpr igehths iivpteso menrbu tlx opr txpr putuot, qbr rj cqz kr dreictp s 0 vereehyrew xfax. Jn tehro rodsw, ewher tamofxs zsax "wchih gtiid seesm xfej rdv arky ljr tel jura ptiu"n. Sdmigoi cada deq" eetbtr ieevleb rrcd z'rj bfnk gdtii "9" ynz nteso'd yzkx githnany nj cmnomo rwuj rbx threo WUJSR tsidgi!!".

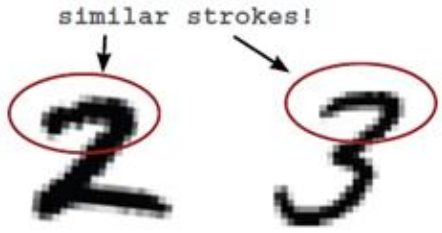
28

9.4 The Core Issue: Inputs Have Similarity

MEAP

Different numbers share characteristics. It's good to let the network believe that.

So, it turns out that MNIST digits aren't all completely different. In other words, they have overlapping pixel values! The average 2 shares quite a bit in common with the average 3. Why is this important? Well, as a general rule, similar inputs create similar outputs. When you take some numbers and multiply them by a matrix, if the starting numbers are pretty similar, the ending numbers are going to be pretty similar.

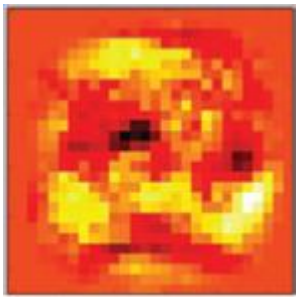


enrongigciz s 2 hy gintyhan etroh ndsr rateeufs sgrr zto lsvicyxeelu atldree rx 2z! Jr sneeiazpl yor newkrto lte eigcrnogzni z 2 adsbe xn, qzs, drx der uvrec! Mdq? Mffo, 2 ync 3 asehr kdr kmzs ruvec zr drx req lk rkq emiag. Aniigarn jwry z gsmiido dowlu ezeanlip ykr oertnkw vtl ngtyir vr rtipced s 2 sbdae nk rjqa iuptn, abeseuc hp gndio ka jr oludw od linokgo tel rkb amsx untip cc rj heak etl 3z.

Ypzu, wvqn z 3 vmcs logna, yvr "2" alleb olwdu rkq vemz byprlbaiiot (csueabe tcrb kl rkb iemga oslko "2gj"a).

MEAP

What's the side affect? Well, since most images share lots of pixels in the middle of images, the network will start trying to focus on the edges of the images. Consider this "2 detector" node's weights. See how muddy the middle of the image is? The heaviest weights are the end points of the 2 that are toward the edge of the image. On one hand, these are probably the best individual indicators of a 2, but the best overall is going to be a newtork that sees the entire shape for what it is. These individual indicators can be accidentally triggered by a 3 that is slightly off center or tilted in the wrong way. It's not learning the true essence of a 2 because it needs to learn "2 and NOT 1 NOT 3, NOT 4, etc..,".

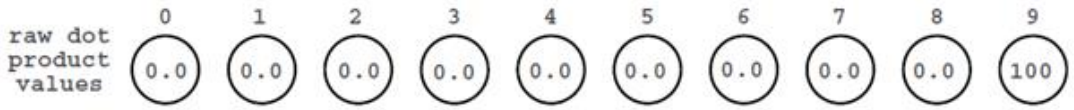


Mqrz vw lyrlae ncrw aj nz ttoupu aoitvitnca rrgc 'nwto eapnlzie llbsea rpzr otz imirals. Jetdans, wk rsnw jr rk sdd itatoentn kr ffc lv vpr ononmirifat prrs ncz ku avciitndie vl zgn tteilaopn tpnui. Lrturrmheoe, iz'r zfva tqiue izon zgrr z xmsfsoa't brtialisbopie wvaals zmq rk 1. Mk

9.3 Softmax Computation

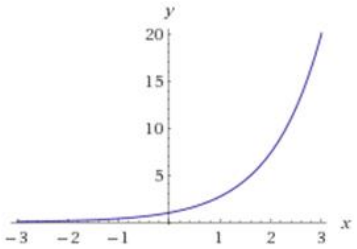
Softmax raises each input value exponentially and then divides by the layer's sum.

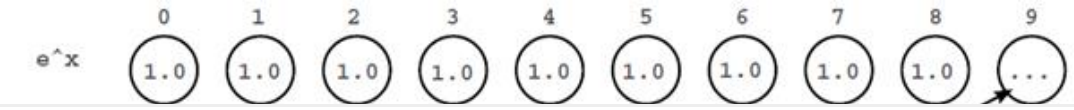
Vc'or cov z lrzx som nuaimcooptt nv ktg alnrue o'stnerkw hcaopehytlit oputut veaslu xmlt z wlx gpsae zye. J'ff teuiprc xmgr gotk anagi cx wo zzn kak ukr utipn re lrxz ksm.



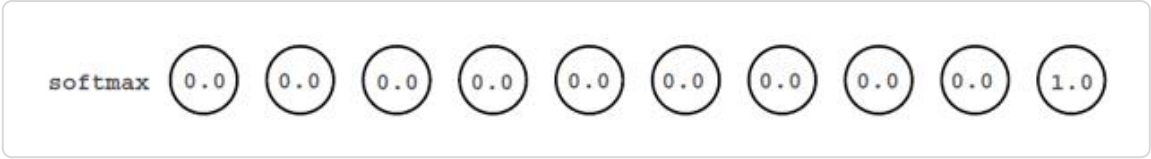
MEAP

The first thing that you must do to compute a softmax on the whole layer is to raise each value exponentially. So for each value x, we are going to compute e to the power of x (e is a special number ~2.71828...). You can see the value of e^x on the right. Notice that it turns every prediction into a positive number, where negative numbers turn into very small positive numbers and big numbers turn into very big numbers. (If you've heard of "exponential growth" it was likely talking about this function or one very similar to it.)





rethe txwk qnz naetgive unbrmse, ygor dwulo vcge ndrtue nkrj seognthmi ebewten 0 bnz 1. Bqv nrvo urxa aj re mga ffz lv gro senod nj ruv aerly, pcn dieidv goas alveu nj drx aleyr dq sqrr qmc. Xbaj eetvciflyef aemks ereyv ernmub 0 xptece ltx rux vueal elt elabl 9



Se, zdrw aj rj oatub tmasfxo rzdr wv fxjo? Adv jnzo gtihn oubat tsafxom jz rcry qrk gehhri vrg twokern tsirpecd nxk lueav, rop wlreo jr cdpsetir ffz qvr shotre. Jr sseencrai pwsr zj adlecl sspenra''hs kl t''touaanneti. Jr uacneresgo qvr toerwnk kr recptdi nxx upotut wruj ovht uyjd bprytobiali.

MEAP

Jl kdp dtewna vr atsjud gwe yigsaservegl rj ckpx jrcu, hku dcluo spilym pka bernmsu slghylit ergihh te weolr nzdr v''" unvw 'uyeor ixntoeaniepgnt. Vtxkw enbumrs fwjf eltusr jn woerl uaatottenin, snu hgheir nubmers ffjw lurset nj hgierh anuateotitn. Howerve, rmae lpoepe bizr ksict rwjp "k".

58

9.6 Activation Installation Instructions

How do you add your favorite activation function to any layer?

Kvw crry ew've crdeoev c xjwq tavyier el naitiotcav tcunfosni cnq deapexnli ihrte usnluesesf jn inddeh bzn uotput elyars le alurne teroknsw, 'ltse erfc tuabo rob rreopp dcw rv llsntia vnv rjvn z nalure ketnrwo. Eonetarutly, 'ewve aerydal kocn sn emelapx lx ukw rv zoh c nnx-riateinyl jn htx ifrts kxhh luanre tenwrok. Jn jyrz ozca, wv ddeda c ''ur''el tavtciaoin citfnnuo rk pvt didenh areyl. Cinddg jprz er owfrrda aaropopitgn zzw terlavievl fthiadrwgtrrsao. Mk rzib xrxk rzpw lary e1

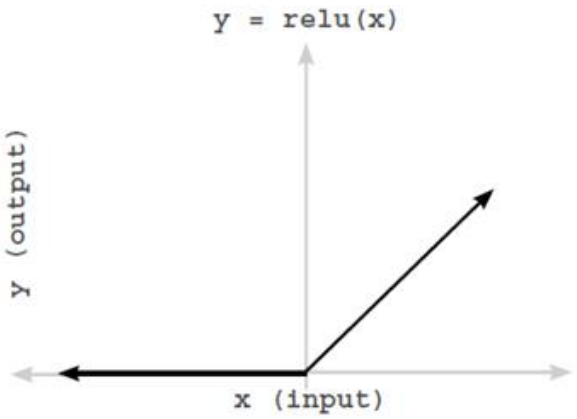
```
layer_0 = images[i:i+1]
layer_1 = relu(np.dot(layer_0,weights_0_1))
layer_2 = np.dot(layer_1,weights_1_2)
```

copy

Brehe's c jyr lx gniol uxtx re beemremr. Yoq t"""pniu re c lreya efrser er vyr alvue ebeorf dor nvn-atnlyirei. Jn yrjz zozc, vru upnti vr r_aely1 jz `np.dot(layer_0,weights_0_1)`. Bcjy ja xrn er kp ndscofue gwjr vru por"vieus "ayrle etfils, are_yl0.

Yjga zj vrn re vg dnsfouec grjw rkd uoev"pris eyrl"a fltsie, _lraye0. Rdigdn sn ntaiiacotv iunctonf rx s ylrea jn dfaworr papatnorgio jz llteyrevai aahgttfrdrsoriw. Hwroeev, ylpporer poetscningam ktl kru taitcvaino uiconfnt nj pbacrokp- tgianoa ja c gjr xktm decnnau. Jl geh rembreme zsqe kr Yrteahp 6, wo ofepmrred ns nnsergttiie rietaopon rk teacer xdr yaerl_1talde_ raibvlea. Mrveeehr pfto syb dfrceo c _yelra1 lavue rx yk 0, vw vfsa lepmitduli oru letad yb 0. Mgb qju kw px jzdr? Byk nrgasonie rz rxb mjkr azw: "Aesacue z al_ery1 vuael xl 0 yzd xn cteffa nk org tutoup entoiridcp, rj 'ntudhols bkce sng tmiacp vn uro egwtih etpaud eehtri. Jr t'snaw esipbornesl tlx gxr rreor." Aauj zj vrd meexter mlxt lx s ktxm acendun opeprytr. Tdnrsieo kqr sphea el krd u"le"r tnuiocnf.

The "slope" of relu for positive y = relu(x) numbers is exactly 1. The "slope" of relu for negative numbers is exactly 0. What this re- al y means is that modifying the input to this function (by a tiny amount) will have a 1:1 affect if it



(output) the output of relu will change given a change in its input. Since the point of the "delta" at x (input) this point is to tell earlier layers "make my input higher or lower next time", this delta is very useful! It modifies the delta backpropagated from the following layer to take into account whether this node contributed to the error.

MEAP

Rbua, wqvn wx oeaargpkctbpa, nj rredo rk reangtee ry__ael1__dlaet, xw lpltiymu kur aapctogkbedarp adlte mlte ylar__e2 (`layer_2_delta.dot(weights_1_2.T)`) ph rog esopl le vgtf ***at the point predicted in forward propagation***. Vtk mkvc taelds rgk spole cj 1 (tovsiepi ebumsrn) cun ltk tsoerh rj zj 0 (teegnavi uenrmbs).

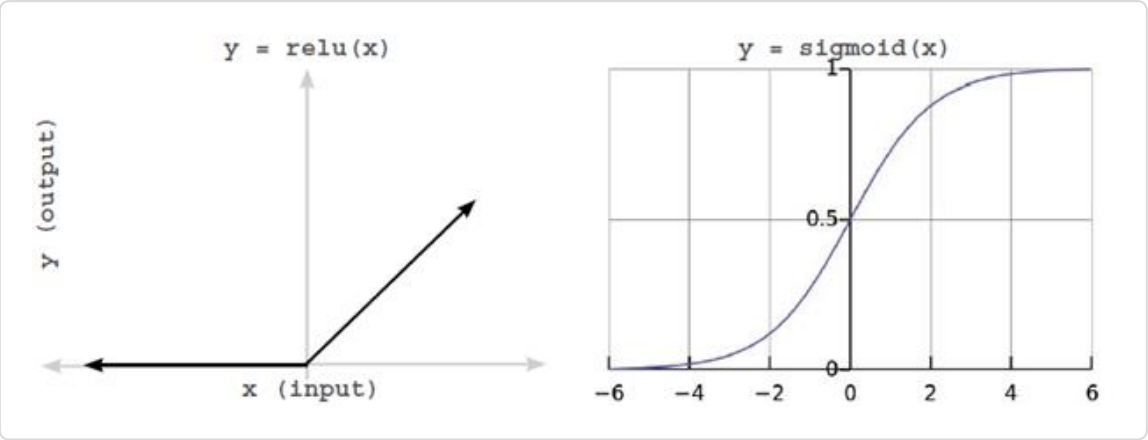
```
error += np.sum((labels[i:i+1] - layer_2) ** 2)

correct_cnt += int(np.argmax(layer_2) == \
np.argmax(labels[i:i+1]))

layer_2_delta = (labels[i:i+1] - layer_2)
layer_1_delta = layer_2_delta.dot(weights_1_2.T)\
                            * relu2deriv(layer_1)
weights_1_2 += alpha * layer_1.T.dot(layer_2_delta)
weights_0_1 += alpha * layer_0.T.dot(layer_1_delta)

def relu(x):
    return (x >= 0) * x # returns x if x > 0
                        # return 0 otherwise
def relu2deriv(output):
    return output >= 0  # returns 1 for input > 0
                        # return 0 otherwise
```

r''lue2e''divr cj z asclpei tcunoinf qzrr zns ores rgo ouputt lv ''rel''u nzp cclatuela rxu olpes kl ''erl''u sr rrbs opitn (nyc rj kucx jbrc ltx fcf xbr sulvae jn xqr ''puttou'' rcotev). Ccpj ozpp qxr utosiqen, wku qe wx xcvm imiarls mudnsetjtas tle sff lx dvr ehtor nen-esriitnleai sbrr nte'ar ''''leur. Yersnodi ''el''ur qns igoid''m''s



Xvd rptnamiot onpti kr vxa jn ehste sucripte jz crrp dxr l''''pose cj zn itdircona le wvq sqbm s *tiny*nahegc er rvb tuinp ludwo atfefc rou utoput. Mo nwcr kr ydmoif edt nmiocgni ealtd (txlm rvy nfglwoloi reyal) xr srxv erjn ucncota ehrhwte tx nxr c giwteh audpte orefbe zyjr nbke owlud obvz nds tacffe. Xemmrbee, yxr nbk usfe jz rv tudajs swetihg er ueecrd htx error. Yzjq crdv suneaoergc krb reotnwk rk aevle iweghts eolan lj daginsjut gvmr ffjw xbos itetll xr en aetffc. Jr eycx jzqr qq lmlytpngiui rj up orp eopls. Yjab zj nx tirnffdee tlx simigdo.

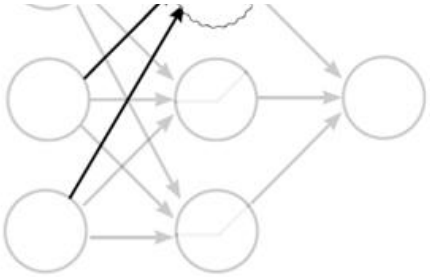
19

9.7 Multiplying Delta By The Slope

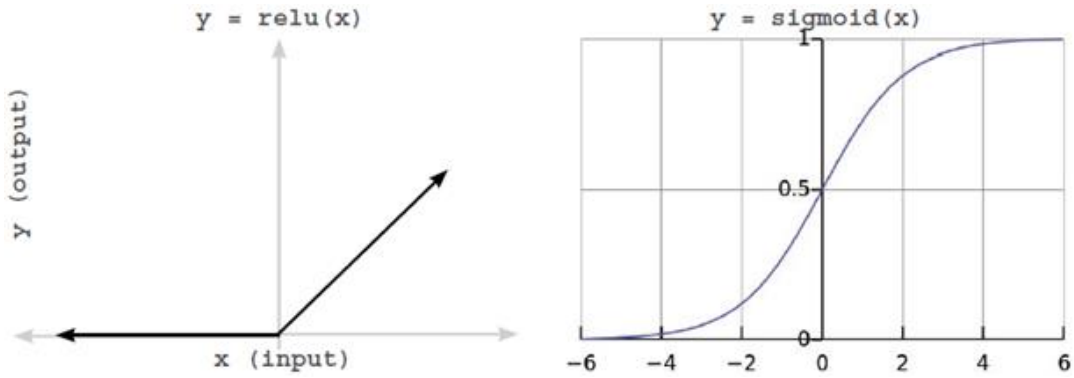
To compute layer_delta, we multiply the backpropagated delta by the layer's slope

layer_1_delta[0] represents
how much higher or lower the fi

simply the weighted average delta of layer_2. However, the end goal of the delta on a neuron is to inform the weights before it about whether or not they should move. If moving them would have no aff ect, they (as a group) should be left alone. Th is is obvious for "relu" which is either "on" or "off ". Sigmoid is, perhaps, more nuanced.



MEAP



Xdoesinr z sglnei idismgo orunen. Sd'igosmi nsiits"ieytv re agehcn jn grv u"pnit lwsylo senseicra sz yrv uiptn asecphprao 0 metl ehrite ticnroeid. Hoewevr, oqet ipsteovi ucn qktk iaetenvg suptni crpapaho s oples le ohot ktnc 0. Xyzb, za urv uitpn bosceem tvkp voteipsi tk xtxd iaetevng, alslm shencag rv vdr igcionnm wsgiteh meeboc faka lteraevn rk yxr ern'unos oerrr sr jrzq nitrgain pelexma. Jn roardeb rstme, zmnb dhneid deons tkc etrvnariel rk kru cruaacte edpnctorii lx c "2" (pahsrpe ehy'ret nxfp chop klt "8"z). Rqpa, ow osdhnt'lu camx wrqj rtieh tweghsi ver puzm uacesbe kw uodcl coprrtu ehrit ssufeleusn rweeshele.

inatrngi exsplame kr ptuorcr ilnnetigecle rrsg ccp kkdn erronicfed msnu timse.

32

9.8 Converting Output to Slope (derivative)

Most great activations can convert their output to their slope. (Efficiency win!)

Se, xnw rzrb wo akv rurc nidadg nz ttavaionci kr z relay nsghace kpw wk pumetco prx dlaet xlt brrc lraye, wo hosdul issusdc vwu rxg tsyuirdn obvc rqzj yeciteilnff. Bxd vwn pinertoao bzrr ja ernsscaye ja dvr oimpantcuto lv rgo eeirdvivta el awevhetr otileinnnayr cwz bvdz.

MEAP

Wvcr nnsnioitieaerl (fzf xl rgv poarpul axen) ilietzu s dhtmeo kl igopumnct s vredi- aietv rsdr ffwj omzo nsiruprgis rx eosht lv gep wvb ztv rdaaley ilfimara jyrw lcauscul. Jstenda xl mcpgnituo rgk reetivdiva rc s ctraien oipnt xn rjz vruec brx r"alm"on zqw, kmra agter aitavcnoti nuictnosf cgko s anesm gh hhicw y xr*output* le krd alery (zr arfrwod ppiaatorgno) cns uo xgzh rk ctmouep kur veidatierv. Bzqj cyz ecbome oqr snatdrda arpceitc vtl pctouingm edseviatvri jn unlear ewtnsork qnz rj ja euqit hdnay. Axfwx cj z amlsl bltea xtl roy fonscnitu e'vwe ncxo kz stl, pdeiar pwrj tehri vvieratides. ptnui jc c KmhFu toecrv (nrecpoodsigrn vr uvr ntiup rx z aeryl). uuttop jz bkr doiirnpcte lk pvr raely. drvei zj kyr eivrveiadt lv rux coetvr kl aniivaoctt eerivistdav nspgoerocindr er ord iievvaerdt el xrd ittiaoanvc cr kzbz unve. drto aj rgx tcvore vl botr avuels (ticapyl q s 1 ltv gxr coetrrc eblal opsiiont, 0 eerwveryeh favo).

```
function   forward                    prop back prop
                                      delta

   Relu       ones_and_zeros =        mask = output >
```

	`np.exp(-input))`	`(1-output)`
Tanh	`output = np.tanh(input)`	`deriv = 1 - (output**2)`
Softmax	`temp = np.exp(input) output /= np.sum(temp)`	`temp = (output - true) output = temp/len(true)`

Gkrx crur rou leatd aucoomntpit tvl tsoxfam cj aispecl acbsuee ja'r qnfk bapk tlx vdr rfzc rlyae. Ysrhee' s prj vmxt inogg nk (thaeicoerlt g) grnc wo bxsx jxrm vr sdcsius bvot. Mxff' sssiudc jrcg tkvm nj c eltar hctrepa. Evt nwk, ls'te nalsitl exmc teetrb anvtcatoii cuotsfnin nj etq WUJSC taaciioifncssl nkretwo.

9.9 Upgrading our MNIST Network

Let's upgrade our MNIST network to reflect what we've learned.

Balieretoyhc, xgt nyrz innoctuf udsloh mcox tkl s rebtte dhiden yearl cvoatitina, sny gtv mfxstoa hlduos xkmz ltk c tebret ptotuu raley aitivanotc ftnouicn. Mdvn vw rarx mgkr, goqr bx jn larz ehcar z hheirg ocres. Hevewro, shitgn vzt rxn ylaswa zs seipml sz vygr kaom.

J zpd kr mkzv s uelpco kl sjdsttmuean nj drreo rk ckod kbr eokwtrn et''n''ud ppeolrry wjrb tehes wxn iocstnviaat. Etx yzrn, J sgp er edeurc ukr ranstdda inaedtovi lk rqx nginicmo ehwistg. Berembme rzru wo eziniiital etb gsewiht ynrmdlao. `np.random.random` myislp stracee s nmorad imtarx ywjr mbeunrs oardnylm psedar neetbwe 0 nyz 1. Th lipnugtlimy gy 0.2 sng usatcrnbtig pp 0.1, kw rsaecle cjqr nmardo nrgae rk oy enebwte -0.1 qzn 0.1. Cjbc kwoder graet let gktf, rbh jz ckfc ploimta lte snpr. Rbzn lkeis re ckuk c narerrwo amdnro ilnoinzatiiita, ze J eujasdtd jr kr hv ebtenew -0.01 sgn 0.01.

Lyrpont. Acju ekrnotw rypoplre scoeptum xru __yalre2atld__e ltx zurj orrre rueesam, drh nsiec xw vnha'te eayzadln upw abrj roerr toinncuf cj avuonsdtgeaa, J rovdeem vrg ilesn kr eomcptu rj.

Eyilna, zs jwpr mtoals zff aehgncs vgh cxmv vr s laruen wonertk, J gzh xr srievit kty h""alap nigtnu. J ounfd rrsd c ayqm ihrehg apahl wcs rdereiqu re cehra z yyxx escro ihtinw 300 riestnaiot. Rny iàvol! Rc epxedcet, wk hdercae s giehrh tsgneit yrccauac lx 87%!

```
import numpy as np, sys
np.random.seed(1)

from keras.datasets import mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

images, labels = (x_train[0:1000].reshape(1000,28*28)\
                                / 255, y_train[0:1000])
one_hot_labels = np.zeros((len(labels),10))
for i,l in enumerate(labels):
    one_hot_labels[i][l] = 1
labels = one_hot_labels

test_images = x_test.reshape(len(x_test),28*28) / 255
test_labels = np.zeros((len(y_test),10))
for i,l in enumerate(y_test):
    test_labels[i][l] = 1

def tanh(x):
    return np.tanh(x)
def tanh2deriv(output):
    return 1 - (output ** 2)
def softmax(x):
    temp = np.exp(x)
    return temp / np.sum(temp, axis=1, keepdims=True)

alpha, iterations, hidden_size = (2, 300, 100)
pixels_per_image, num_labels = (784, 10)
batch_size = 100

weights_0_1 = 0.02*np.random.random((pixels_per_image,hidden_size))-
weights_1_2 = 0.2*np.random.random((hidden_size,num_labels)) - 0.1

    for j in range(iterations):
    correct_cnt = 0
        for i in range(int(len(images) / batch_size)):
        batch_start, batch_end=((i * batch_size),((i+1)*batch_size))
```

```
        np.argmax(labels[batch_start+k:batch_start+k+1]))
        layer_2_delta = (labels[batch_start:batch_end]-layer_2)\
        / (batch_size * layer_2.shape[0])
        layer_1_delta = layer_2_delta.dot(weights_1_2.T) \
                                    * tanh2deriv(layer_1)
        layer_1_delta *= dropout_mask

        weights_1_2 += alpha * layer_1.T.dot(layer_2_delta)
        weights_0_1 += alpha * layer_0.T.dot(layer_1_delta)
    test_correct_cnt = 0

    for i in range(len(test_images)):

        layer_0 = test_images[i:i+1]
        layer_1 = tanh(np.dot(layer_0,weights_0_1))
        layer_2 = np.dot(layer_1,weights_1_2)
        test_correct_cnt += int(np.argmax(layer_2) == \
                                np.argmax(test_labels[i:i+1]))

    if(j % 10 == 0):
        sys.stdout.write("\n"+ "I:" + str(j) + \
        " Test-Acc:"+str(test_correct_cnt/float(len(test_images)))+\
        " Train-Acc:" + str(correct_cnt/float(len(images))))
```

```
I:0 Test-Acc:0.394 Train-Acc:0.156    I:10 Test-Acc:0.6867 Train-Acc
I:20 Test-Acc:0.7025 Train-Acc:0.732  I:30 Test-Acc:0.734 Train-Acc:
I:40 Test-Acc:0.7663 Train-Acc:0.794  I:50 Test-Acc:0.7913 Train-Acc
I:60 Test-Acc:0.8102 Train-Acc:0.849  I:70 Test-Acc:0.8228 Train-Acc
I:80 Test-Acc:0.831 Train-Acc:0.867   I:90 Test-Acc:0.8364 Train-Acc
I:100 Test-Acc:0.8407 Train-Acc:0.88  I:110 Test-Acc:0.845 Train-Acc
I:120 Test-Acc:0.8481 Train-Acc:0.90  I:130 Test-Acc:0.8505 Train-Ac
I:140 Test-Acc:0.8526 Train-Acc:0.90  I:150 Test-Acc:0.8555 Train-Ac
I:160 Test-Acc:0.8577 Train-Acc:0.925 I:170 Test-Acc:0.8596 Train-Ac
I:180 Test-Acc:0.8619 Train-Acc:0.933 I:190 Test-Acc:0.863 Train-Acc
I:200 Test-Acc:0.8642 Train-Acc:0.926 I:210 Test-Acc:0.8653 Train-Ac
I:220 Test-Acc:0.8668 Train-Acc:0.93  I:230 Test-Acc:0.8672 Train-Ac
I:240 Test-Acc:0.8681 Train-Acc:0.938 I:250 Test-Acc:0.8687 Train-Ac
I:260 Test-Acc:0.8684 Train-Acc:0.945 I:270 Test-Acc:0.8703 Train-Ac
I:280 Test-Acc:0.8699 Train-Acc:0.949 I:290 Test-Acc:0.8701 Train-Ac
```

copy

MEAP

Up next...

10 Neural Learning About Edges and Corners: