Multiclass Classification

Deep Learning

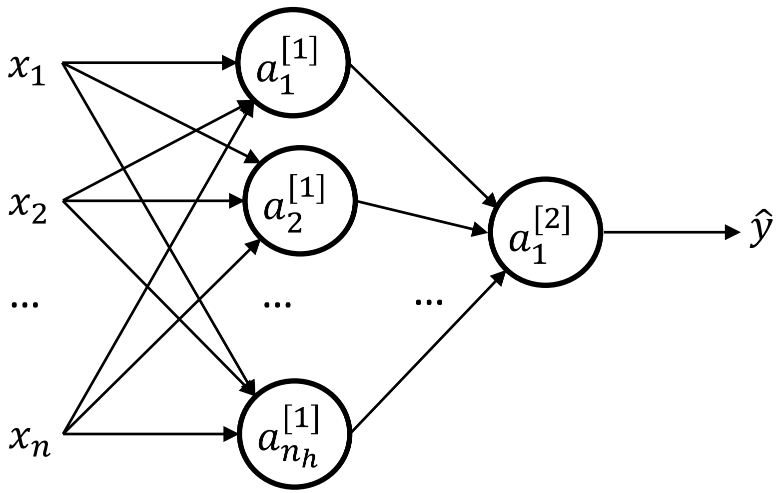
Brad Quinton, Scott Chin

Learning Objectives

- The Multiclass Classification Problem
- How to encode the output for a Neural Network
- Common approaches to Multiclass Classification
- Softmax Activation Function
- Categorical Cross-Entropy Loss
- Back Propagation through Softmax Layer

Quick Recap

- Models
  - Logistic Regression
  - 2-Layer Neural Network
  - A model consists of its architecture and parameter values



$$Z^{[2]} = g\left(W^{[2]}A^{[1]} + B^{[2]}\right)$$

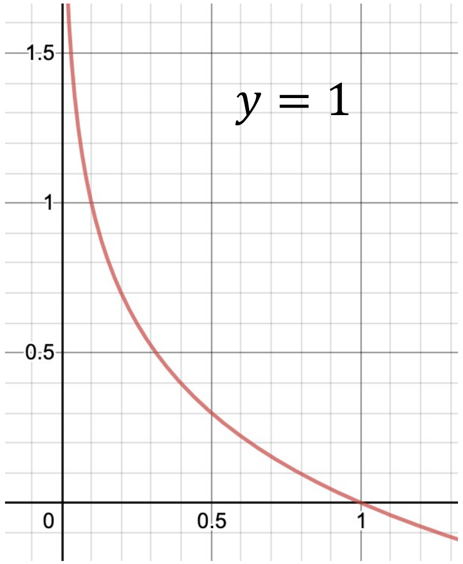
Brad Quinton, Scott Chin

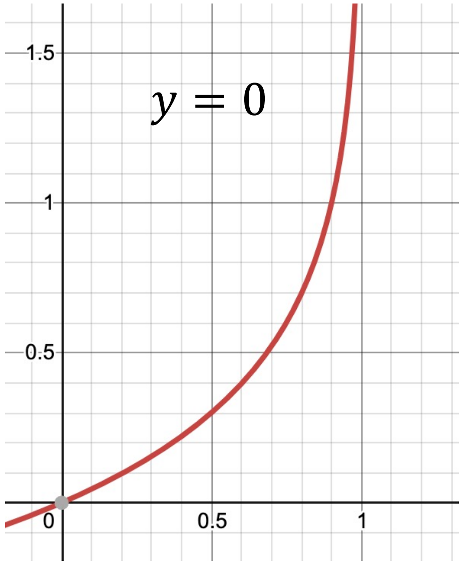
Quick Recap

- Loss
  - Numerical measure of how good the model's prediction is on a single example
  - Low means prediction is close (correct), high means prediction is far away (wrong)
  - Binary Cross-Entropy Loss (aka Log Loss, Logistic Loss)

$$L(\hat{y}, y) = -(y\log(\hat{y}) + (1-y)\log(1-\hat{y}))$$

$$L(\hat{y}, y) = \begin{cases} -\log(\hat{y}), & y = 1 \\ -\log(1-\hat{y}), & y = 0 \end{cases}$$





Brad Quinton, Scott Chin

Quick Recap

- Training (for Supervised Learning)
  - Data has known labels (e.g. the expected output of your model)
  - Goal is to find good values for model parameters from labeled examples.
  - Use Cost Function (aka Objective Function) to measure how good the current parameters are

- Training Cost Function
  - Minimize average Loss across **<u>all training samples</u>**

$$J(W,B) = -\frac{1}{m}\sum_{i=1}^{m} L\left(y^{(i)}, \hat{y}^{(i)}\right)$$

Brad Quinton, Scott Chin

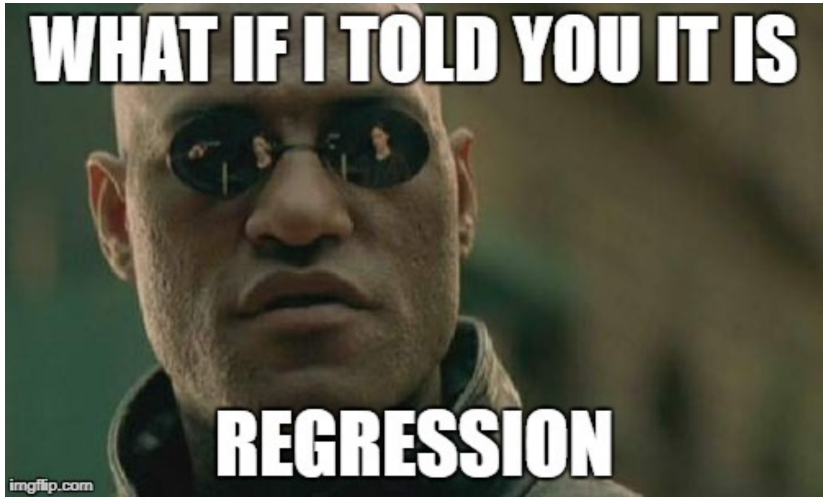
Quick Recap

- Gradient Descent
  - Use Gradient Descent to iteratively search for parameter values that minimize the Cost Function
  - Back propagation key enabler to finding partial derivatives needed for Gradient Descent in Neural Networks

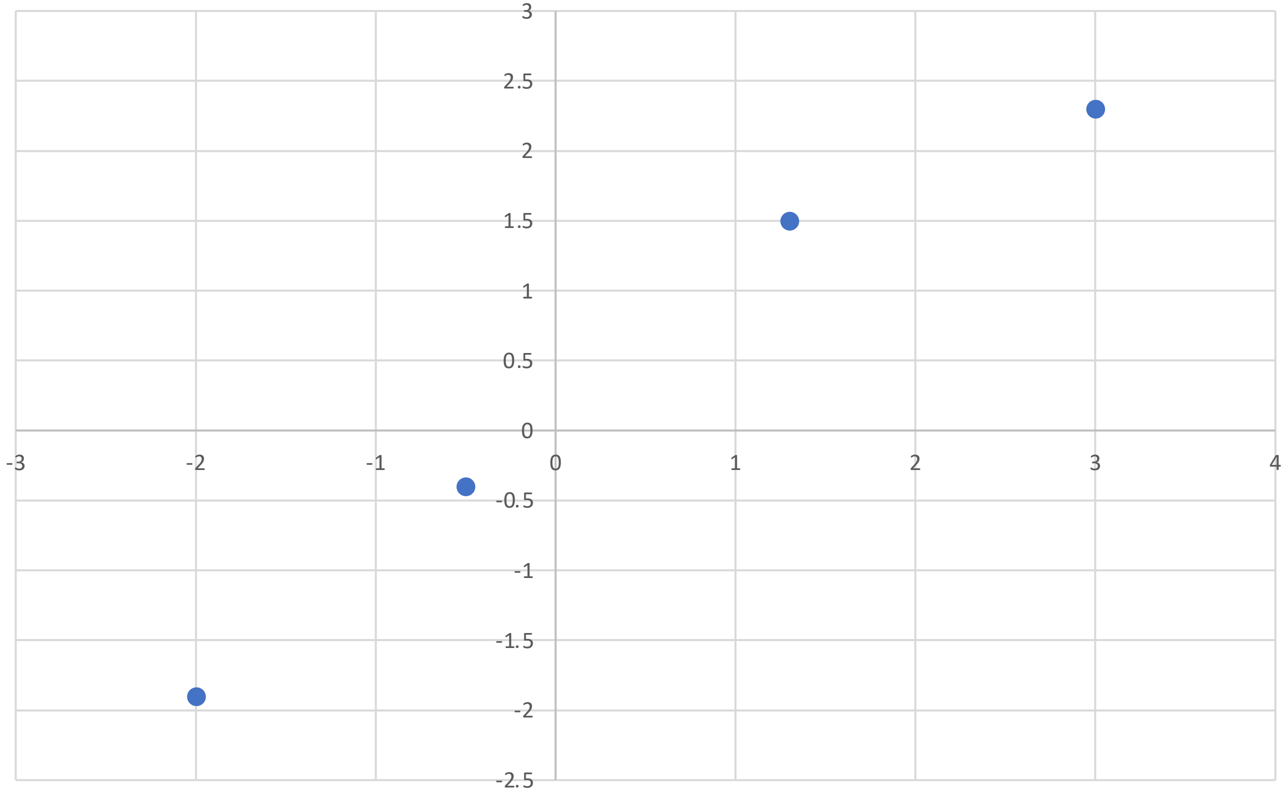
- Training
  - Training/Validation/Test data split to properly assess model
  - Overfitting

There is no magic!

I’m sure you’ve all done machine learning!



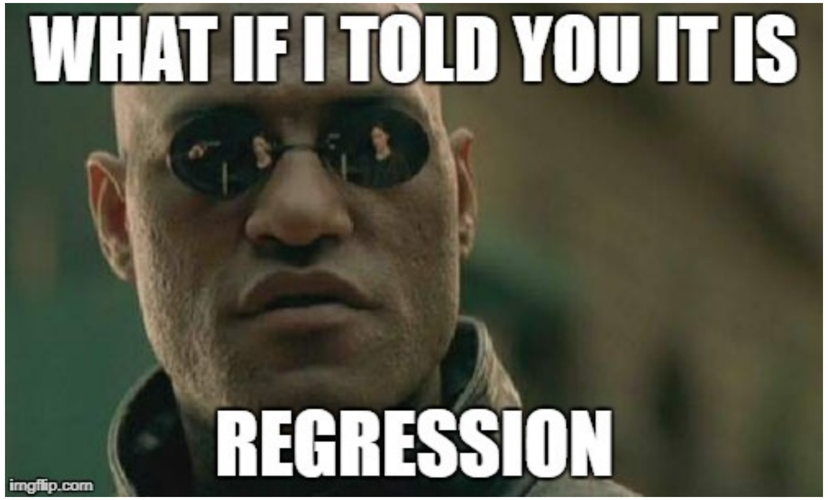
X	Y
-2	-1.9
-0.5	-0.4
1.3	1.5
3	2.3



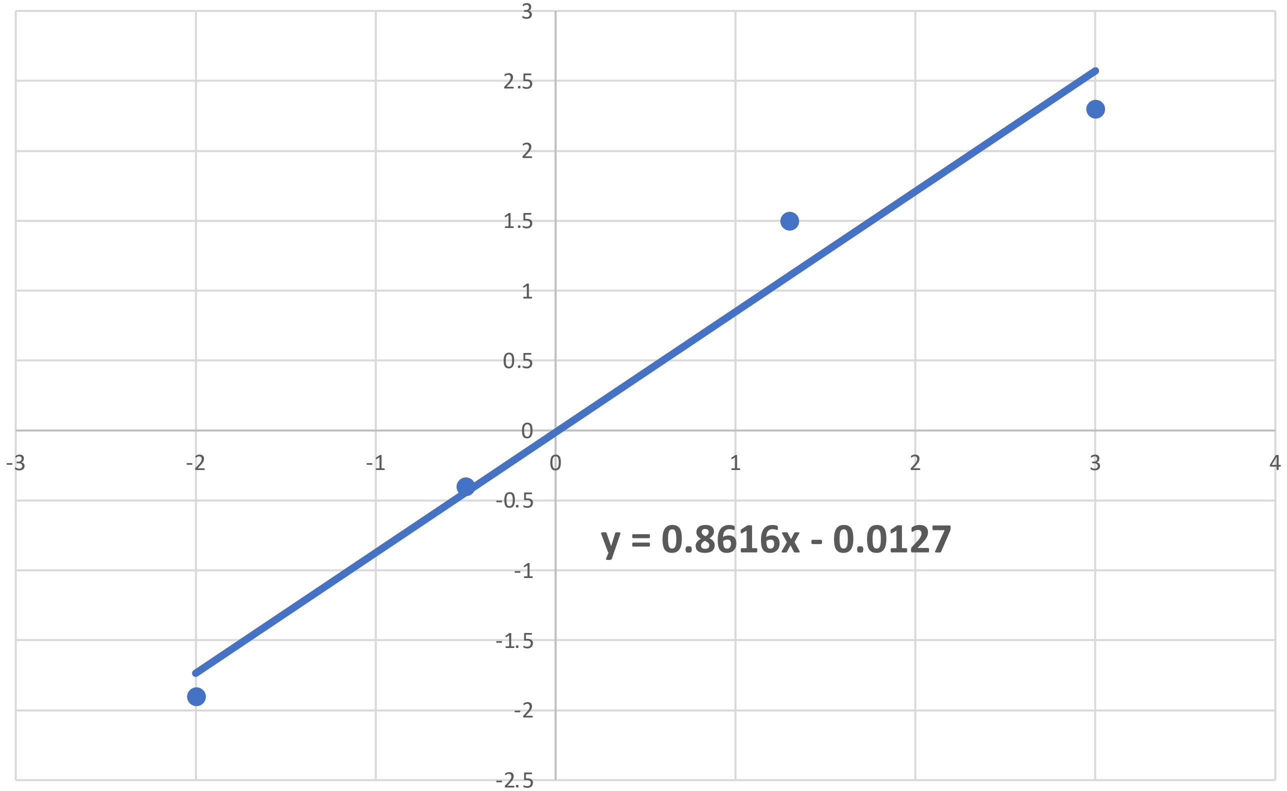
Brad Quinton, Scott Chin

There is no magic!

I'm sure you've all done machine learning!
Excel Add Trendline! y = mx + b



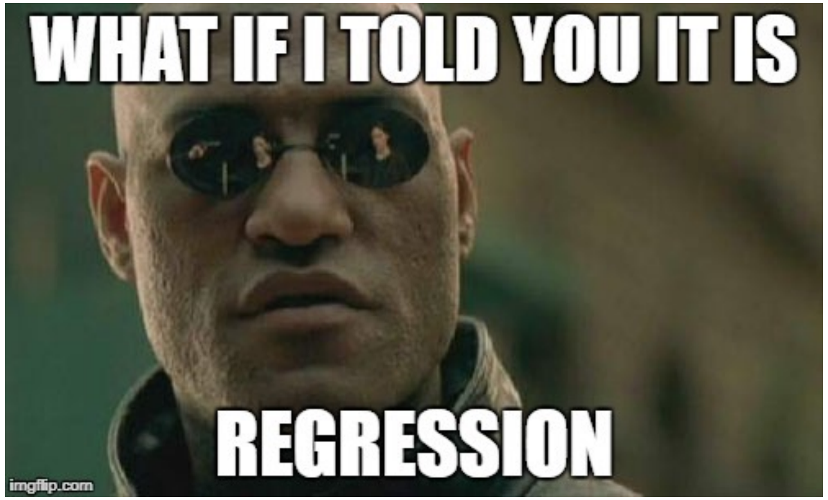
X	Y
-2	-1.9
-0.5	-0.4
1.3	1.5
3	2.3



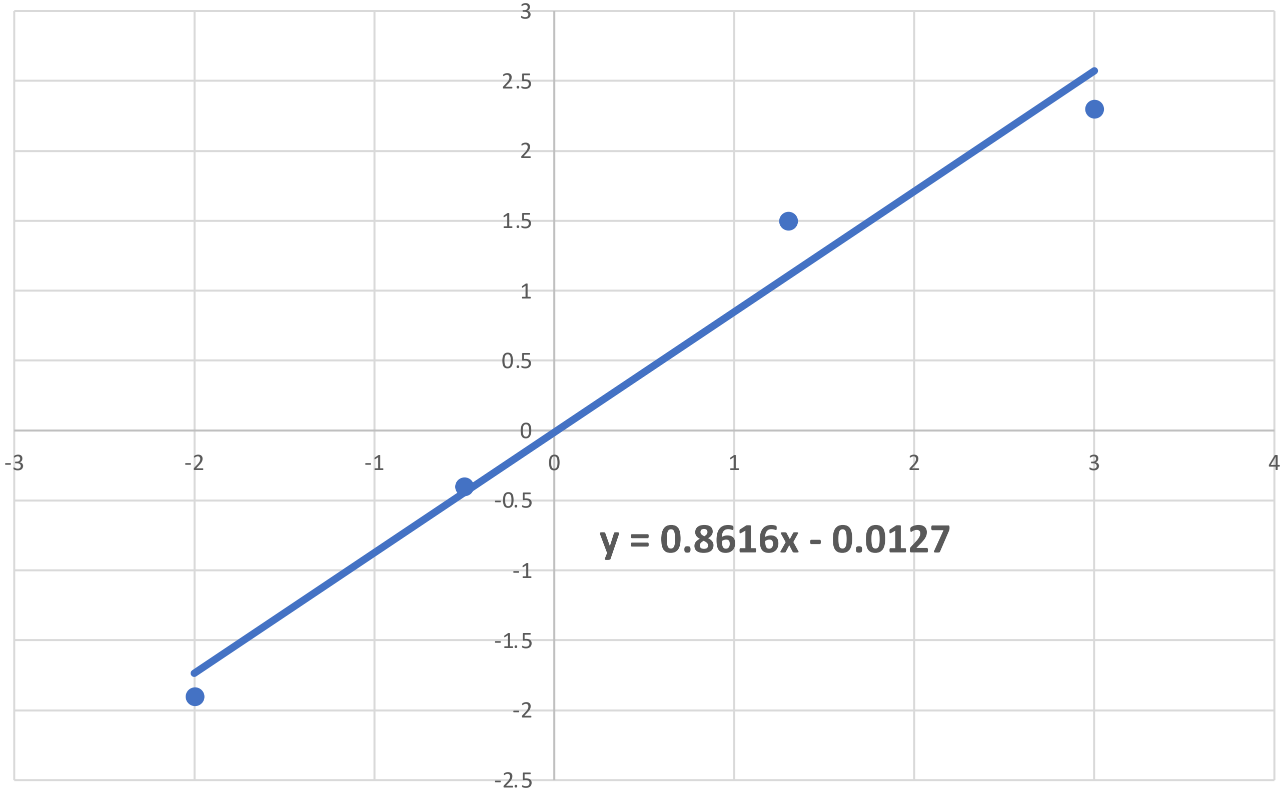
Brad Quinton, Scott Chin

There is no magic!

I’m sure you’ve all done machine learning!
Excel Add Trendline! y = mx + b



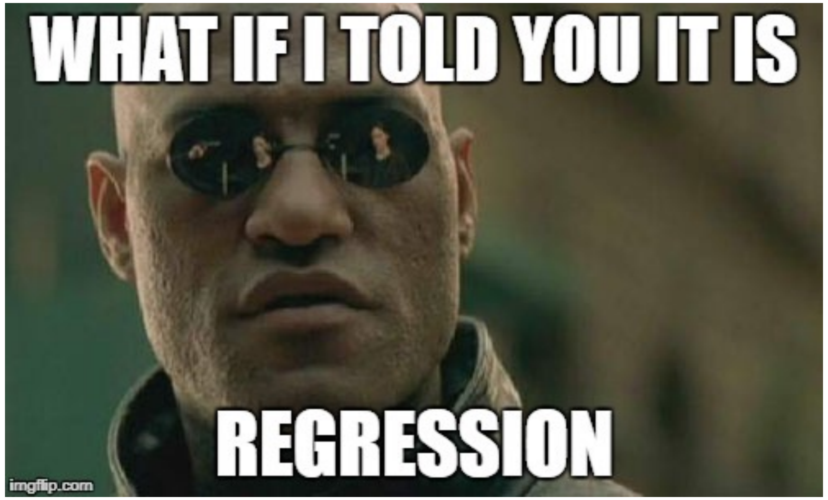
X	Y	Y_hat
-2	-1.9	-1.73
-0.5	-0.4	-0.44
1.3	1.5	1.11
3	2.3	2.57



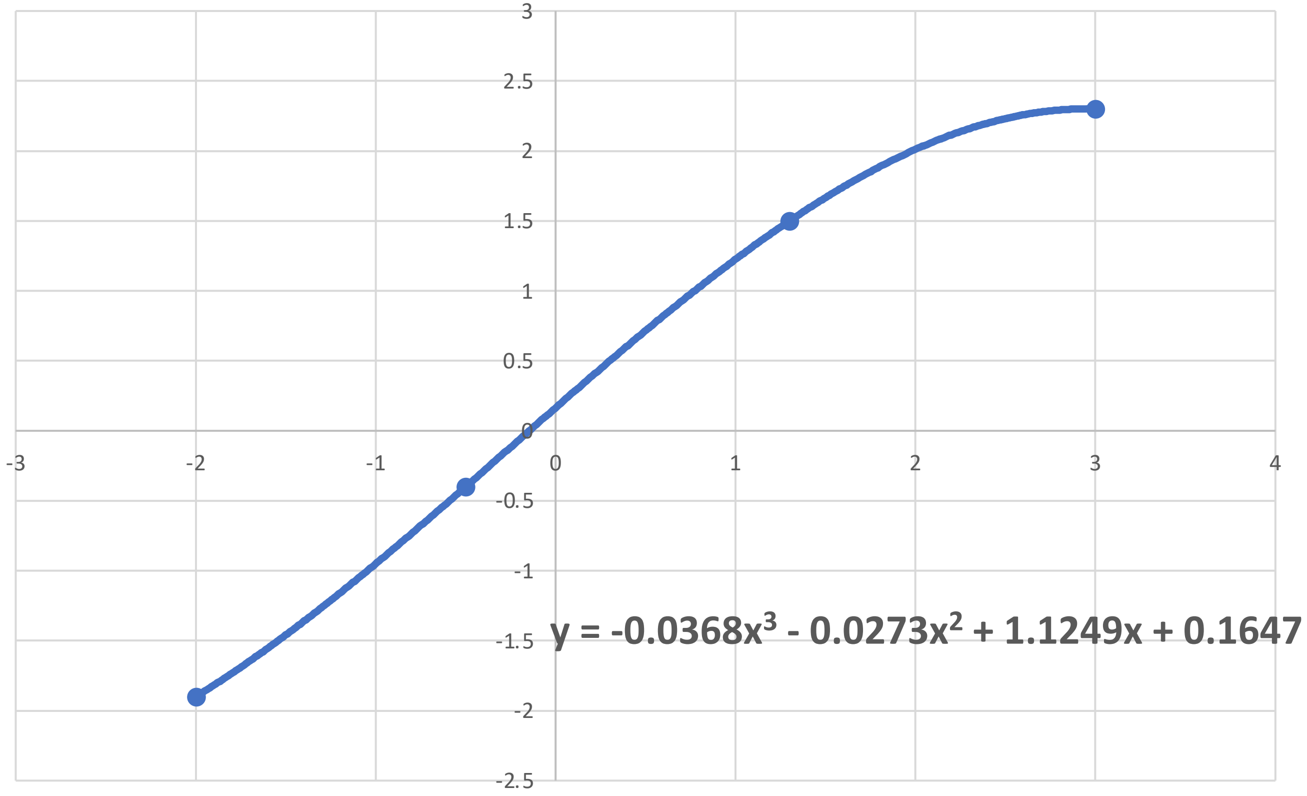
Brad Quinton, Scott Chin

There is no magic!

I'm sure you've all done machine learning!
Excel Add Trendline! $y = m_3x^3 + m_2x^2 + m_1x^1 + b$



X	Y	Y_hat
-2	-1.9	-1.9
-0.5	-0.4	-0.4
1.3	1.5	1.5
3	2.3	2.3



Brad Quinton, Scott Chin

Recap – There is no magic!

Difference from the Excel example and what we will look at using Deep Learning:

- Input is more than one variable (i.e. feature) . Tens of thousands
- Model is more sophisticated
- Model has any more parameters. 100 million+

Caveat: Machine Learning and Deep Learning are broad fields, and we definitely should not say it is all just curve fitting. But it is a useful analogy for stepping into the fundamentals that we focus on in the coming weeks.

Brad Quinton, Scott Chin

Recap - Binary Classification

- So far, we've talked about logistic regression and neural networks that predict a 0 or 1. Input is classified into two possible **classes.**
- The Classification Problem can be used to model directly, or be a key building block to modelling many real world problems.

Is this a cat?

Model

1
(yes)

Brad Quinton, Scott Chin

Aside – Images as Input Data

How do we supply an image as input to our model?

Aside - Images as Input Data

- A grayscale image can be modelled as an array of pixels
- Each array value is from 0-255 representing brightness of the pixel
- 0 for black, 255 for white, and grays in between

H

W

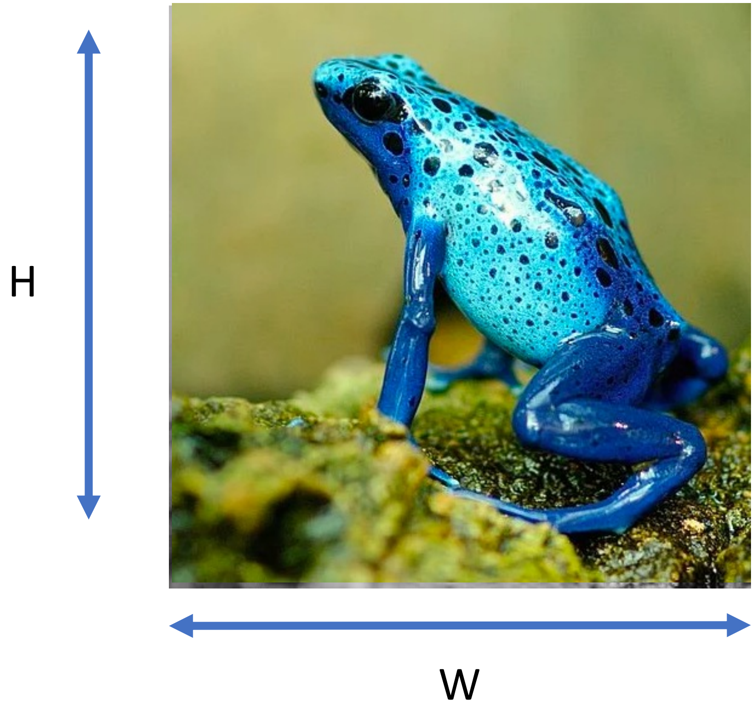
H

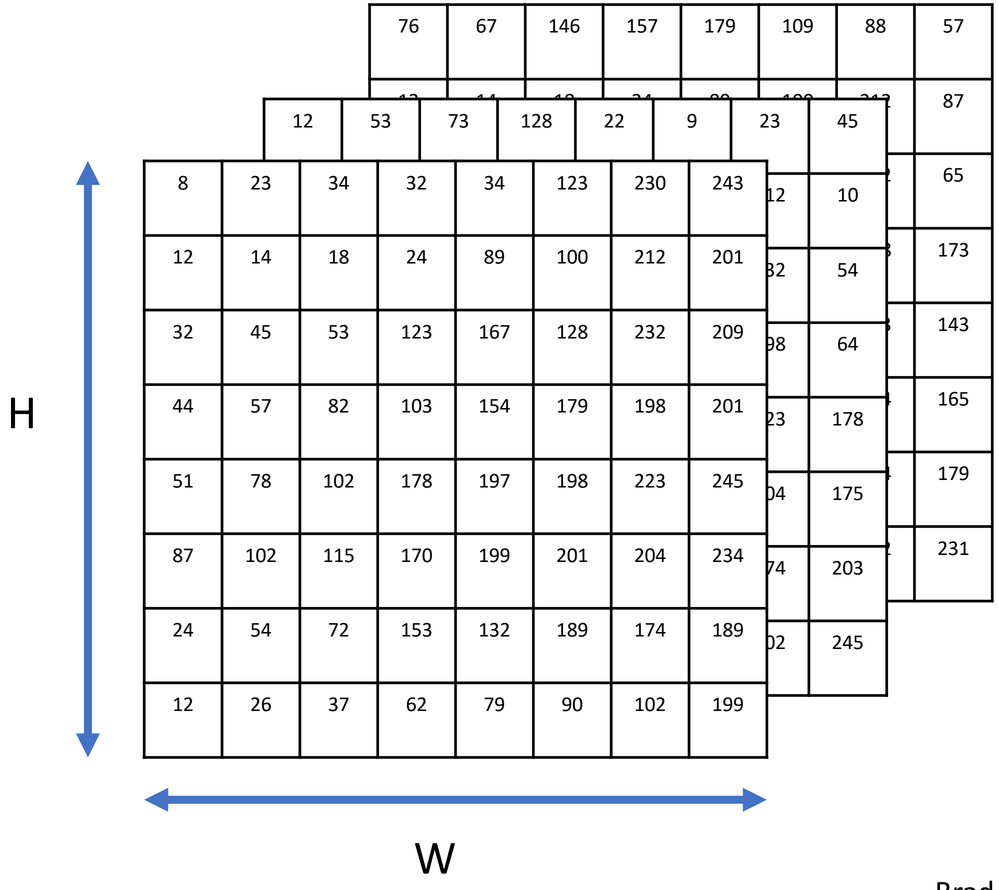
8	23	34	32	34	123	230	243
12	14	18	24	89	100	212	201
32	45	53	123	167	128	232	209
44	57	82	103	154	179	198	201
51	78	102	178	197	198	223	245
87	102	115	170	199	201	204	234
24	54	72	153	132	189	174	189
12	26	37	62	79	90	102	199

W

Aside - Images as Input Data

- For color image, model as three channels (RGB) → HxWx3
- Each array value is still 0-255
- R=255, G=153, B=0 → Orange



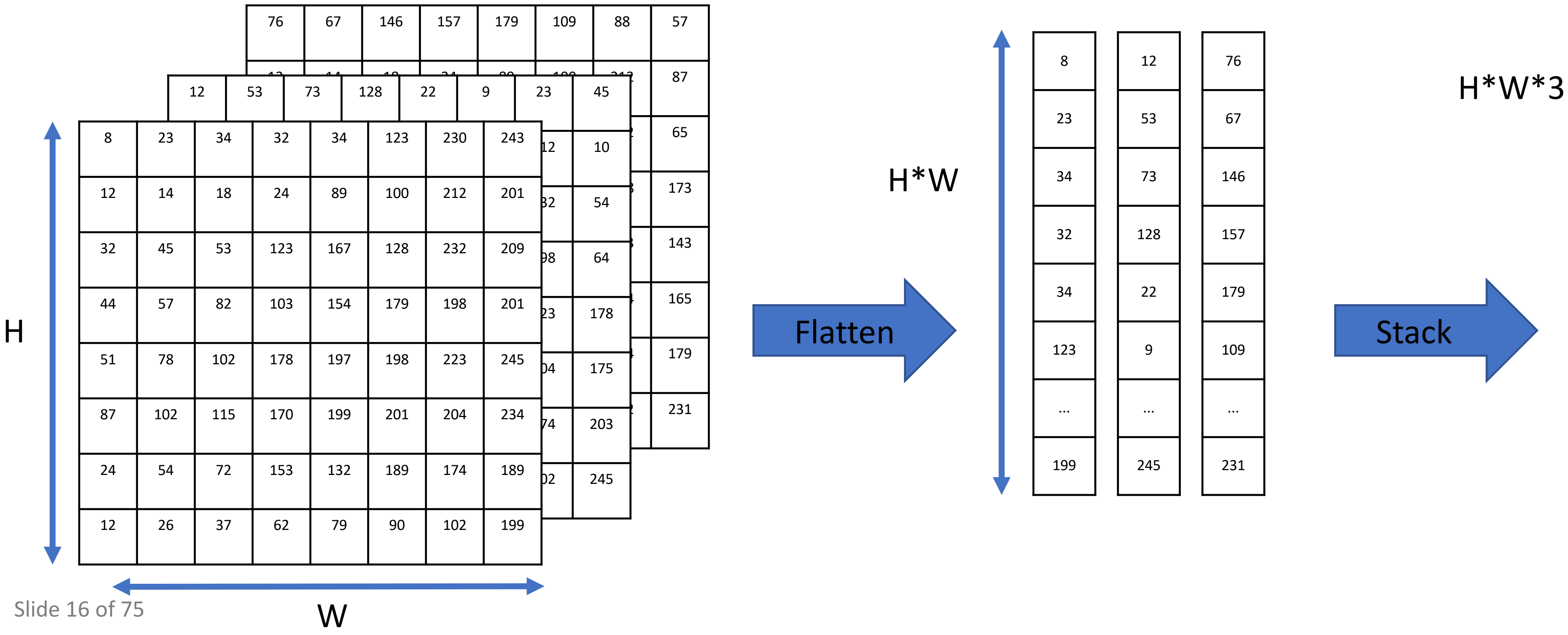


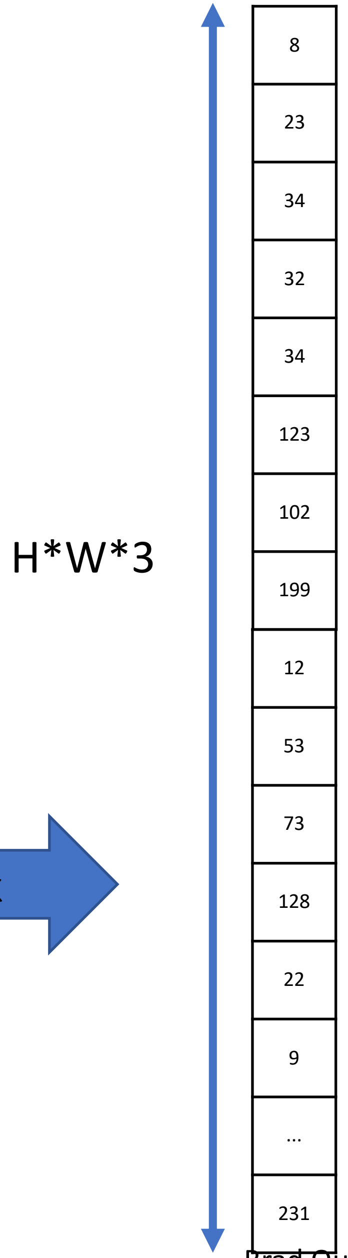
8	23	34	32	34	123	230	243
12	14	18	24	89	100	212	201
32	45	53	123	167	128	232	209
44	57	82	103	154	179	198	201
51	78	102	178	197	198	223	245
87	102	115	170	199	201	204	234
24	54	72	153	132	189	174	189
12	26	37	62	79	90	102	199

Brad Quinton, Scott Chin

Convert to a feature vector

- Flatten each array into a vector and concatenate
- HxWx3 array becomes a (H*W*3) vector





Brad Quinton, Scott Chin

Aside - Image as a vector

- Each pixel is a feature
- Can use with Logistic Regression and Neural Networks we’ve seen
- Later, we will see Convolutional Neural Networks and won’t need to convert to a feature vector

Model

0
(is not a cat)

Brad Quinton, Scott Chin

Back to Binary Classification

Binary Classification – Examples

- Is this email spam or not?
- Is this tumor malignant or benign
- If we (ex. google) show this ad to this person, will they click it?

Some problems can be modelled this way

Brad Quinton, Scott Chin

Binary Classification – Examples

- Is this email spam or not?
- Is this tumor malignant or benign
- If we (ex. google) show this ad to this person, will they click it?

But for many problems we want to classify into more than two classes

Brad Quinton, Scott Chin

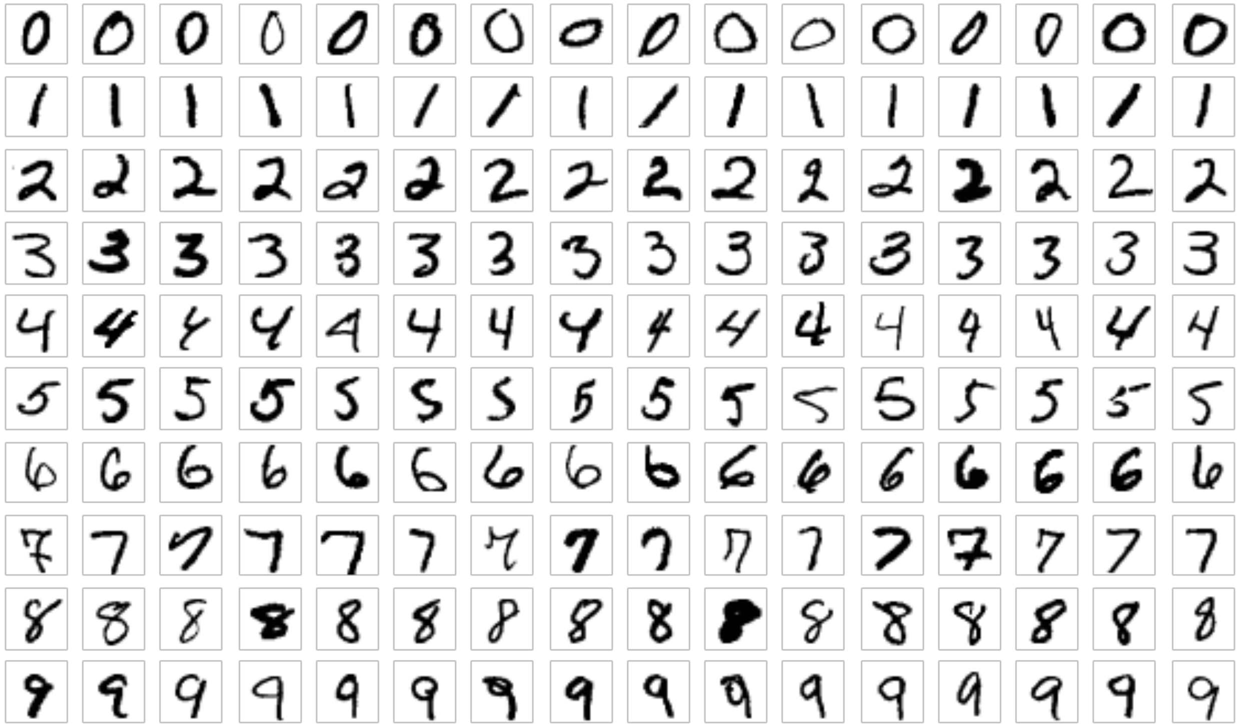
Multiclass Classification

- number of possible classes n_c
  - $n_c = 2$ for the binary classification

Brad Quinton, Scott Chin

Multiclass Classification

- number of possible classes n_c
  - $n_c = 2$ for the binary classification
- Which handwritten digit is this?
  - 10 classes – MNIST dataset



Brad Quinton, Scott Chin

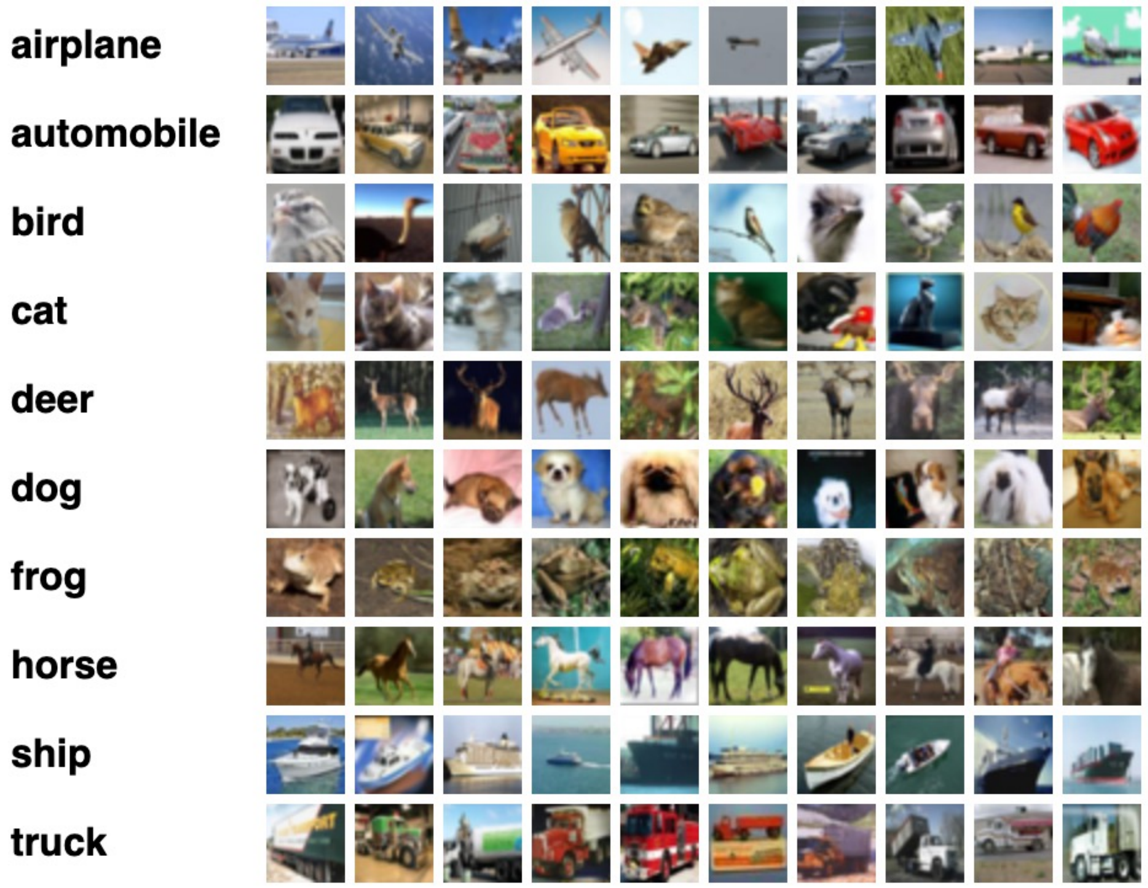
Multiclass Classification

- number of possible classes n_c
  - $n_c = 2$ for the binary classification

- Which handwritten digit is this?
  - 10 classes – MNIST dataset
- What is this a picture of out of 20,000 possibilities?
  - ImageNet dataset

Multiclass Classification

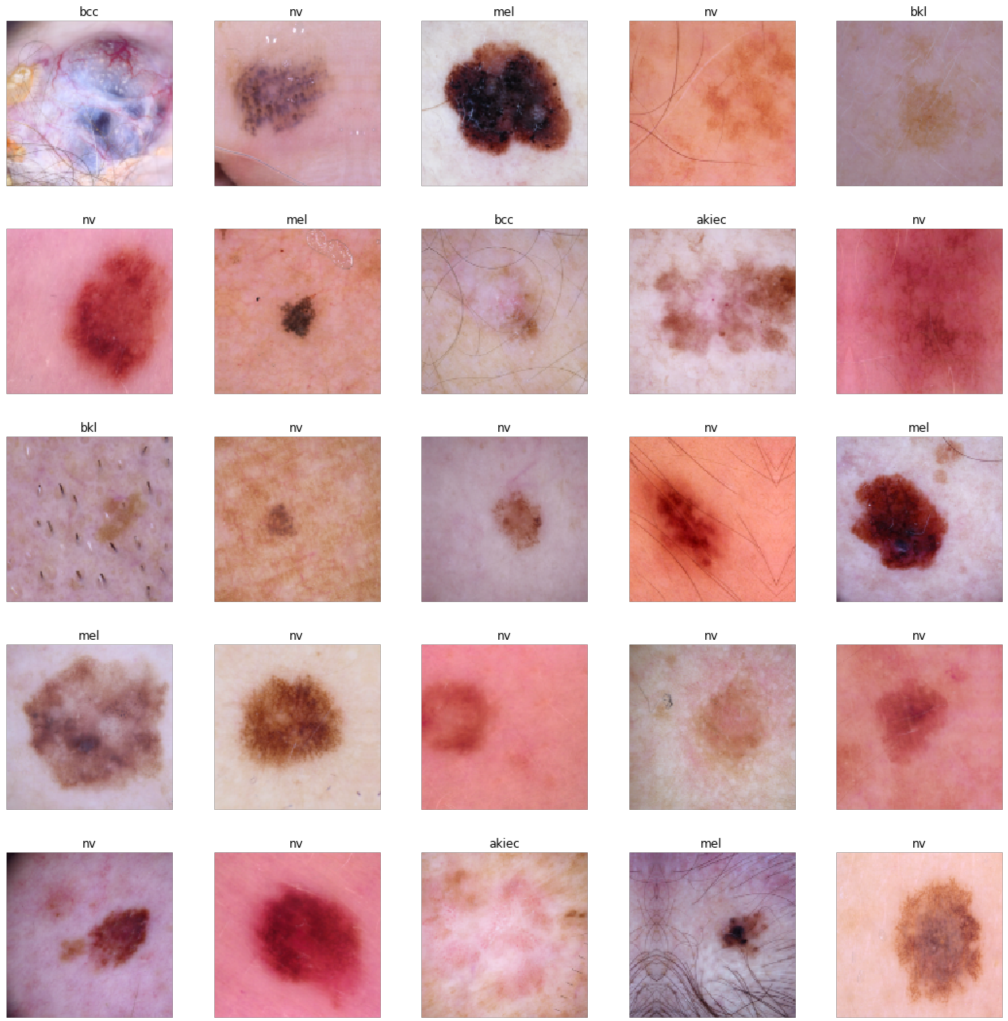
- number of possible classes n_c
  - $n_c = 2$ for the binary classification
- Which handwritten digit is this?
  - 10 classes – MNIST dataset
- What is this a picture of out of 20,000 possibilities?
  - ImageNet dataset
- What is this a picture of out of 10 possibilities?
  - CIFAR10 dataset



Brad Quinton, Scott Chin

Multiclass Classification

- number of possible classes n_c
  - $n_c = 2$ for the binary classification
- Which handwritten digit is this?
  - 10 classes – MNIST dataset
- What is this a picture of out of 20,000 possibilities?
  - ImageNet dataset
- What is this a picture of out of 10 possibilities?
  - CIFAR10 dataset
- Which of 9 skin cancers is this a picture of?
  - ISIC Dataset



Brad Quinton, Scott Chin

Multiclass vs. Multilabel

- Multiclass Classification
  - Input has exactly one label

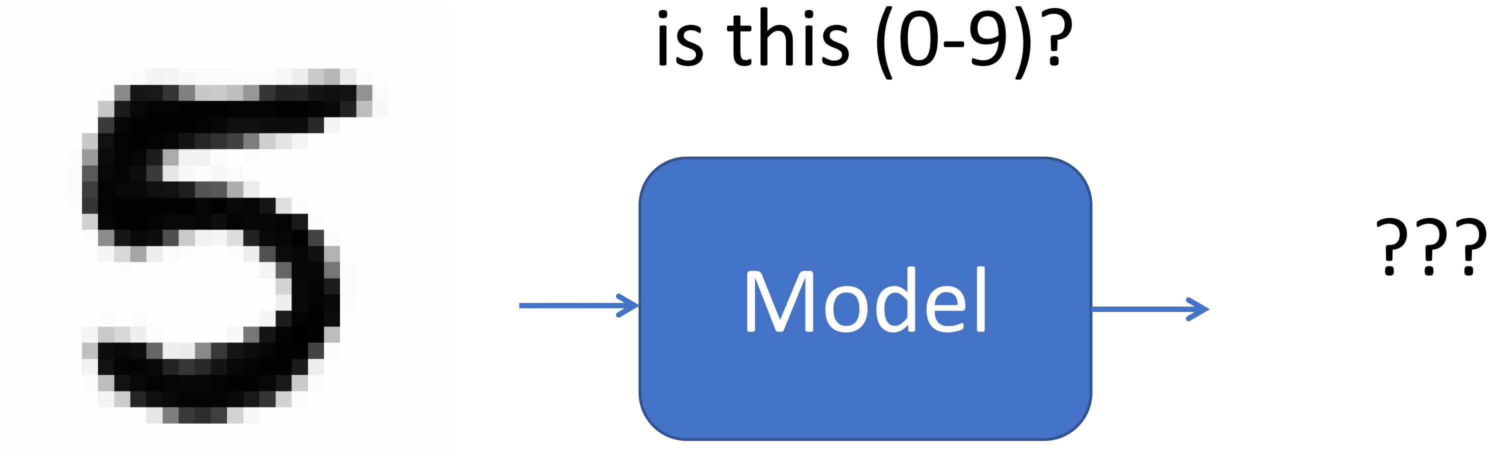
- Multilabel Classification
  - Input has one or more label
  - Examples:
    - News articles → Topic(s)
    - Movie poster → Movie genre(s)

For now, we will look at Multiclass Classification

Multiclass Classification

- How can we encode the output?
  - i.e. instead of 0 and 1, what should our model output?

Which digit
is this (0-9)?

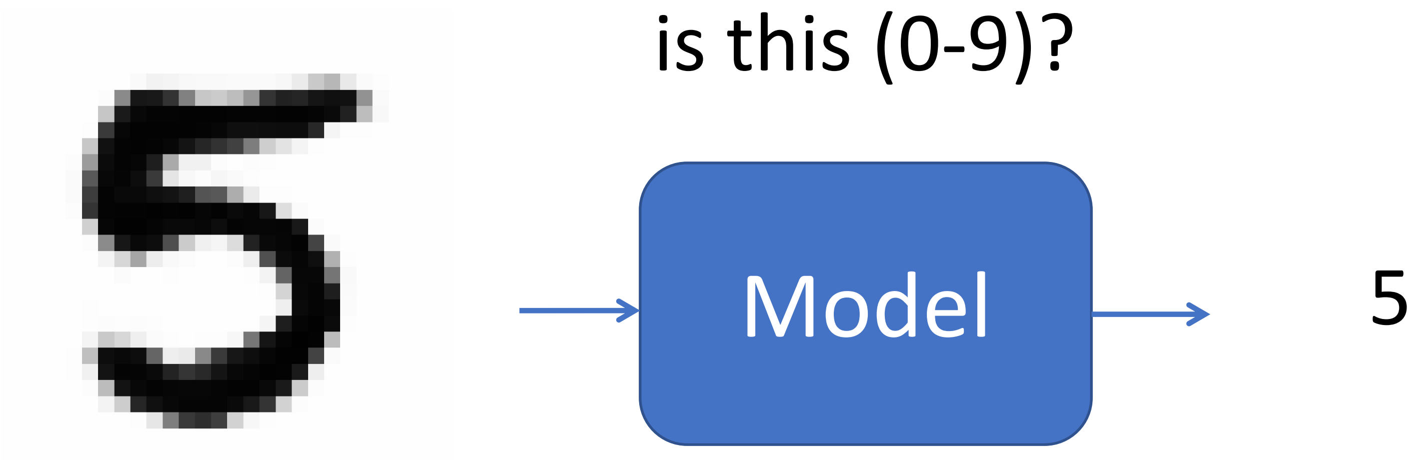


Brad Quinton, Scott Chin

Multiclass Classification

- How can we encode the output?
  - i.e. instead of 0 and 1, what should our model output?
  - One possibility: A single number ranging from 0 to 9?

Which digit
is this (0-9)?



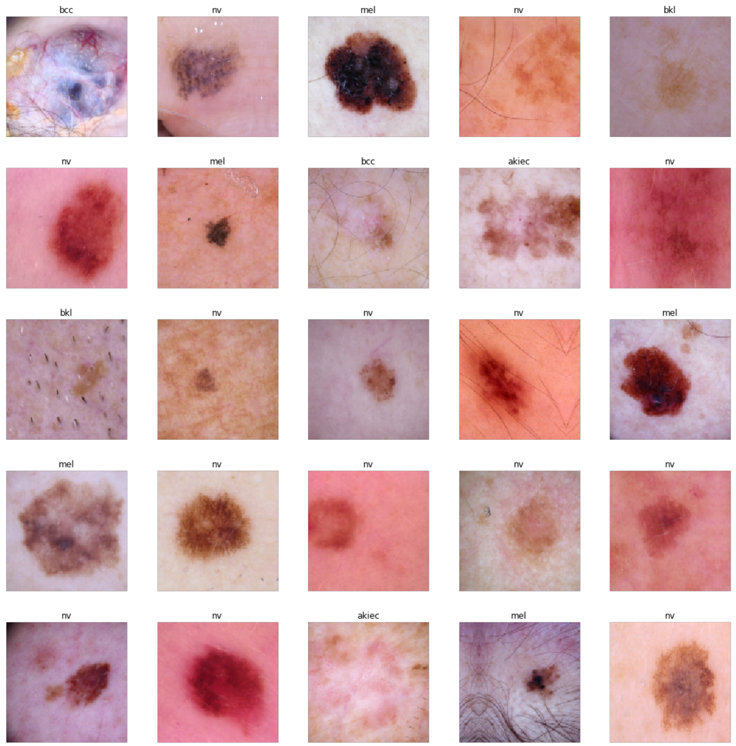
Brad Quinton, Scott Chin

Output Discrete Range of Values

- How to map discrete categories to the integer values?

Output Discrete Range of Values

- How to map discrete categories to a single continuous output?
  - Example: ISIC Dataset has 9 classes:
    1. Melanoma
    2. Melanocytic nevus
    3. Basal cell carcinoma
    4. Actinic keratosis
    5. Benign keratosis
    6. Dermatofibroma
    7. Vascular lesion
    8. Squamous cell carcinoma
    9. None of the others

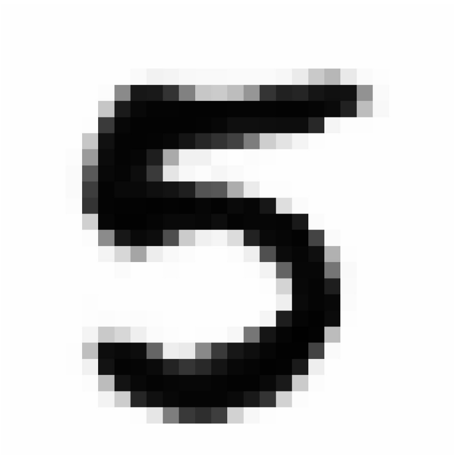


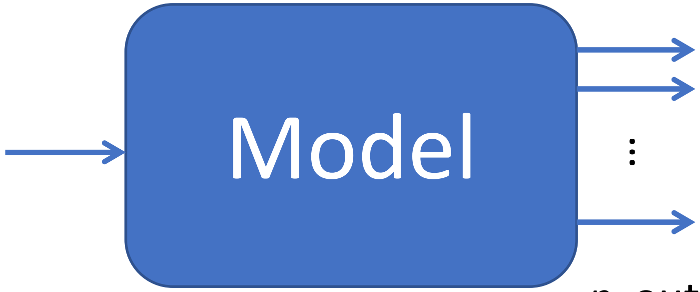
Brad Quinton, Scott Chin

Multiclass Classification

- One-hot encoded vector of length n_c

Which digit is this (0-9)?





n_c outputs

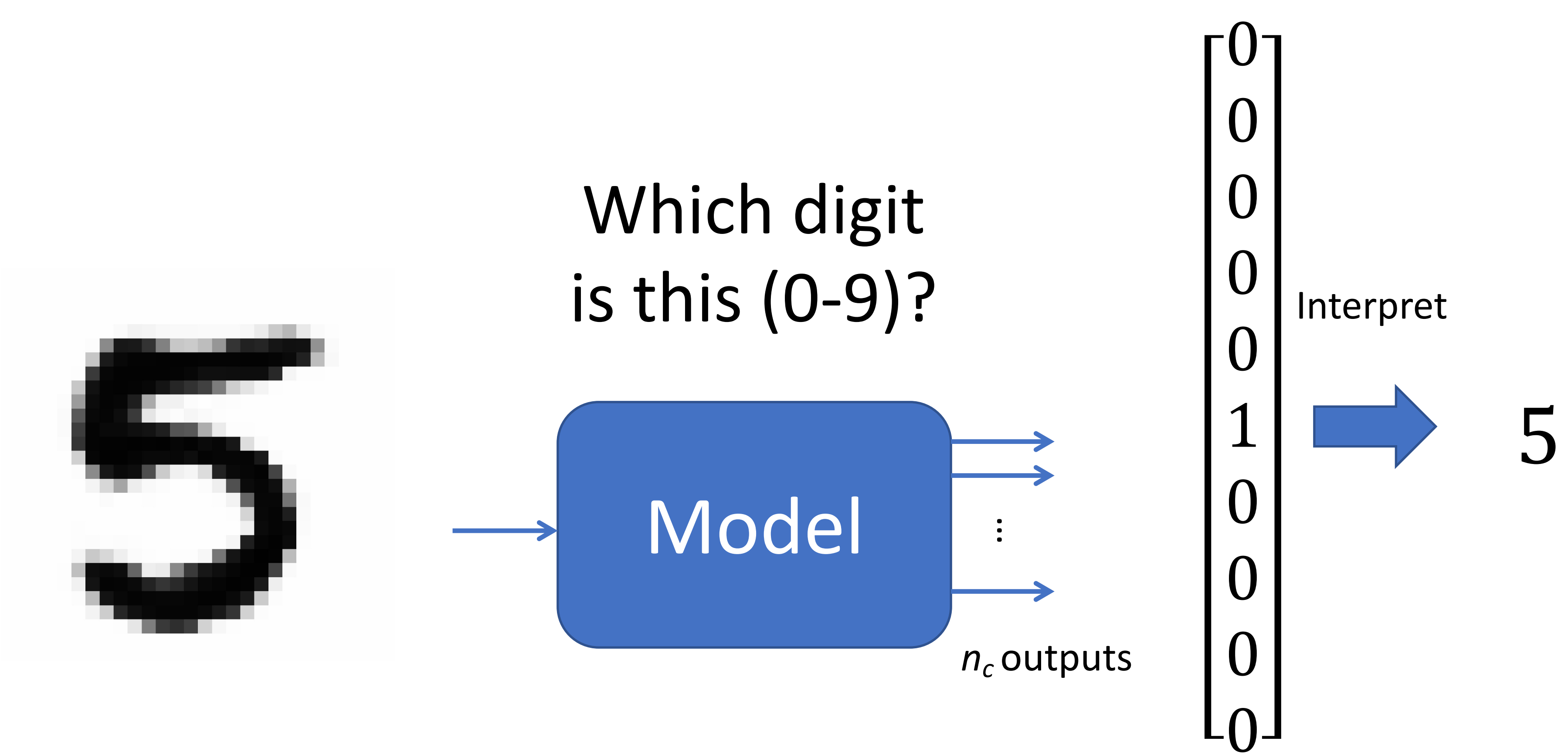
Output

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Brad Quinton, Scott Chin

Multiclass Classification

- One-hot encoded vector of length n_c



Brad Quinton, Scott Chin

Why One-Hot Encoding?

Brad Quinton, Scott Chin

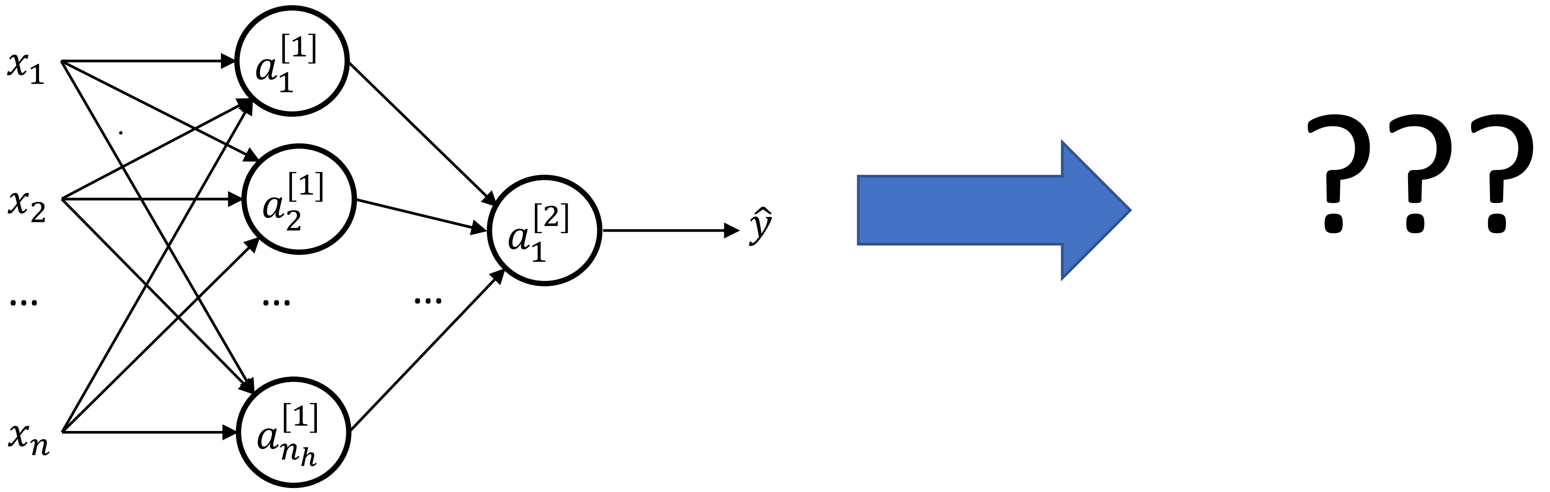
Another advantage of One-Hot Encoding

- Allows us to extend what we know already about building Binary Classification models (Both Logistic Regression and Neural Networks)!

Brad Quinton, Scott Chin

How to extend Binary Classifier to Multiclass

- Let's say we have n_c classes, How we can we extend our binary classifier?



Brad Quinton, Scott Chin

Common Approaches

- Multiple Binary Classifiers
  - One-vs-All (a.k.a. One-vs-Rest)
  - One-vs-One

- Single Classifier With Multiple Outputs
  - This approached used with deep neural networks

Brad Quinton, Scott Chin

One-vs-All (aka One-vs-Rest)

- Build multiple binary classifiers
- One binary classifier per class
- Each classifier predicts whether the input is in its class or not

Brad Quinton, Scott Chin

One-vs-All Example

- Say you are building image classifier for Simpson characters

  0 - Homer

  1 - Ned Flanders

  2 - Moe Szyslak

  …

  19 Mayor Quimby

- One binary classifier for each.
  - Homer classifier trained with data where Homer pictures are labelled as 1, and pictures of all other characters are labelled as 0
  - Ned Flanders classifier trained with data where Ned Flander pictures are labelled as 1, and pictures of all other characters are labelled as 0
  - …

One-vs-One

- Build $n_c(n_c-1)/2$ binary classifiers
  - i.e. all possible combinations of 2 classes
  - Homer vs Ned
  - Homer vs Moe
  - Homer vs Lisa
  - …
  - Ned vs Moe
  - Ned vs Lisa
  - …

- Training
  - Each classifier only receives data about the pair of classes it is discriminating between
- Prediction/Inference
  - Use a majority voting scheme to select the class that was predicted the most often amongst the $n_c(n_c-1)$ binary classifiers

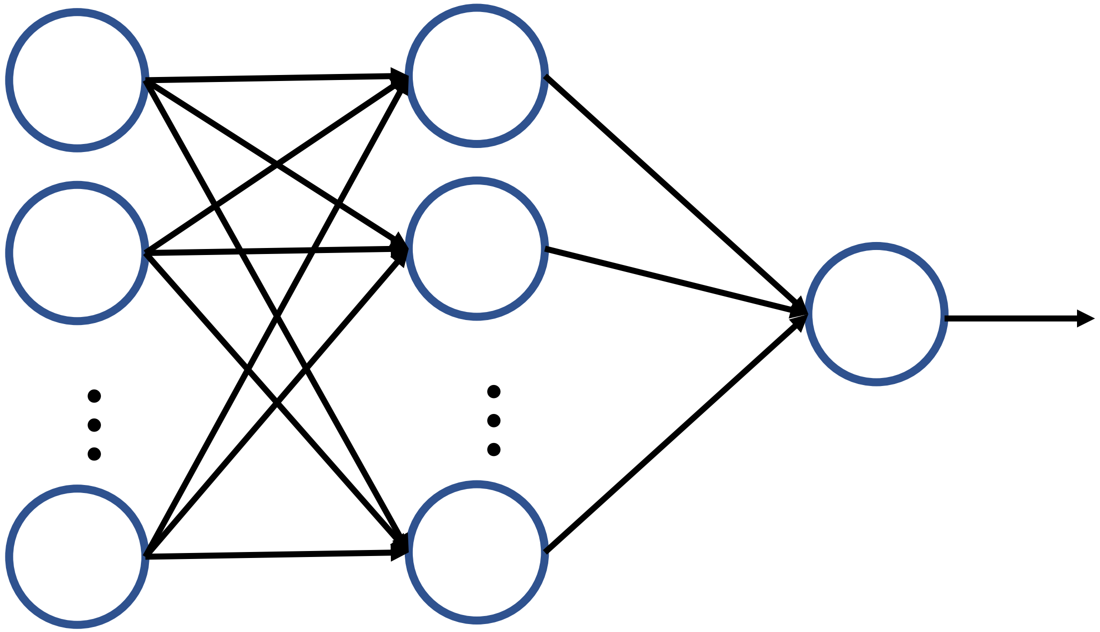
One vs One (OvO) vs One vs All (OvA)

- OvO scales poorly with number of classes
  - 5 classes → need 10 binary classifiers
  - 10 classes → need 45 binary classifiers
  - 100 classes → need 4950 binary classifiers
- OvO and OvA perform about the same. Anecdotally, I've read that people find that OvO can do a little better

- In Deep Learning (Deep Neural Networks) is generally both more efficient to train and performs better

Single Neural Network with Multiple Outputs

- One neural network like before,



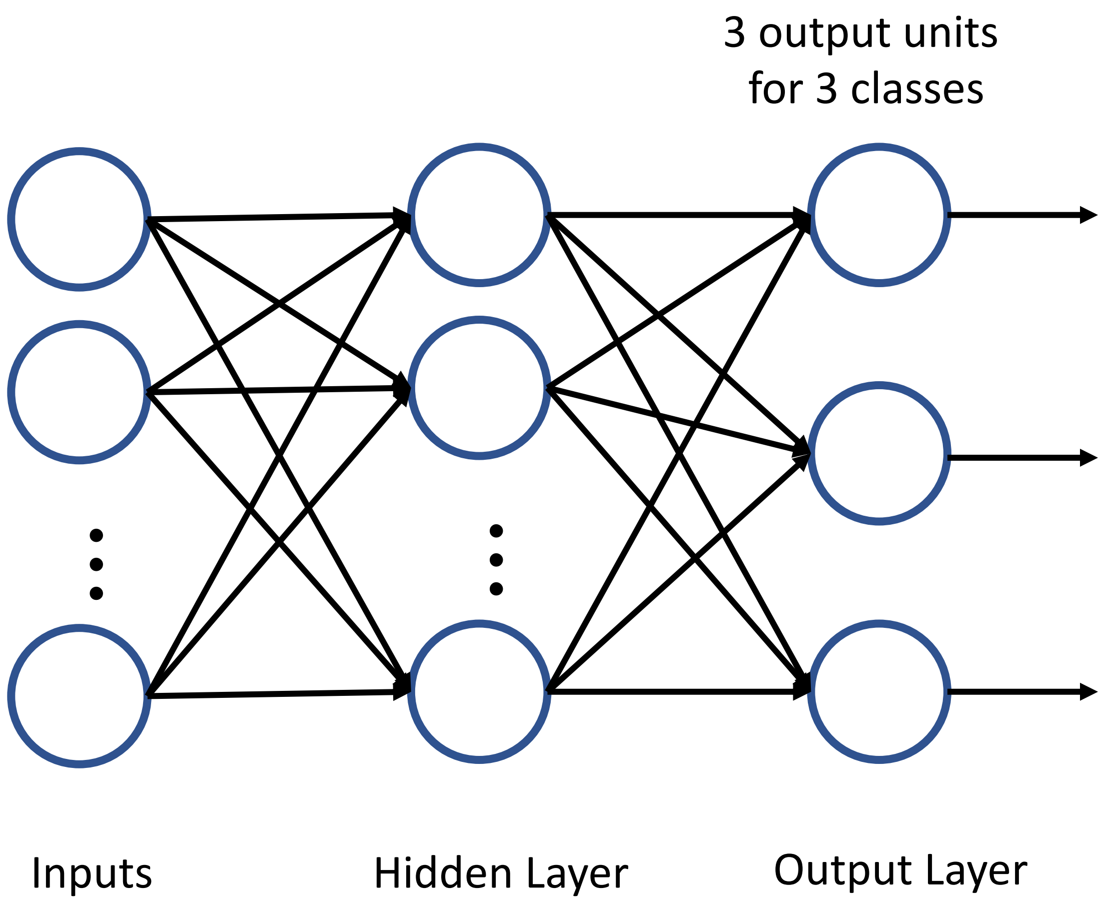
Notes:

- This works the same for Logistic Regression
- For rest of the course, we will refer to this kind of multiclass neural networks unless otherwise noted

Brad Quinton, Scott Chin

Single Neural Network with Multiple Outputs

- One neural network like before,
- Change output layer to have one node per class.
- Each output continues to act as a binary classifier for that class (i.e. predicts a 0 or 1)



Notes:

- This works the same for Logistic Regression
- For rest of the course, we will refer to this kind of multiclass neural networks unless otherwise noted

Brad Quinton, Scott Chin

Multinomial vs One-vs-Rest

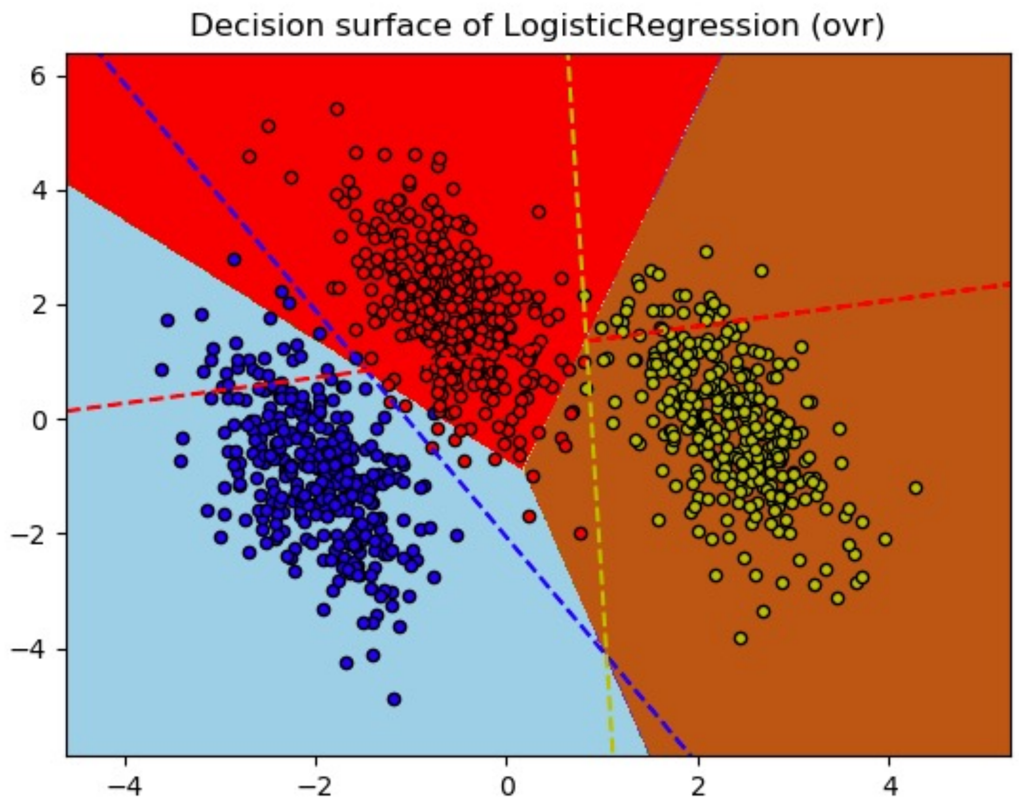
Both have n_c output nodes

Multinomial

- Classes are mutually exclusive
- See shaded regions in figure

One-vs-Rest

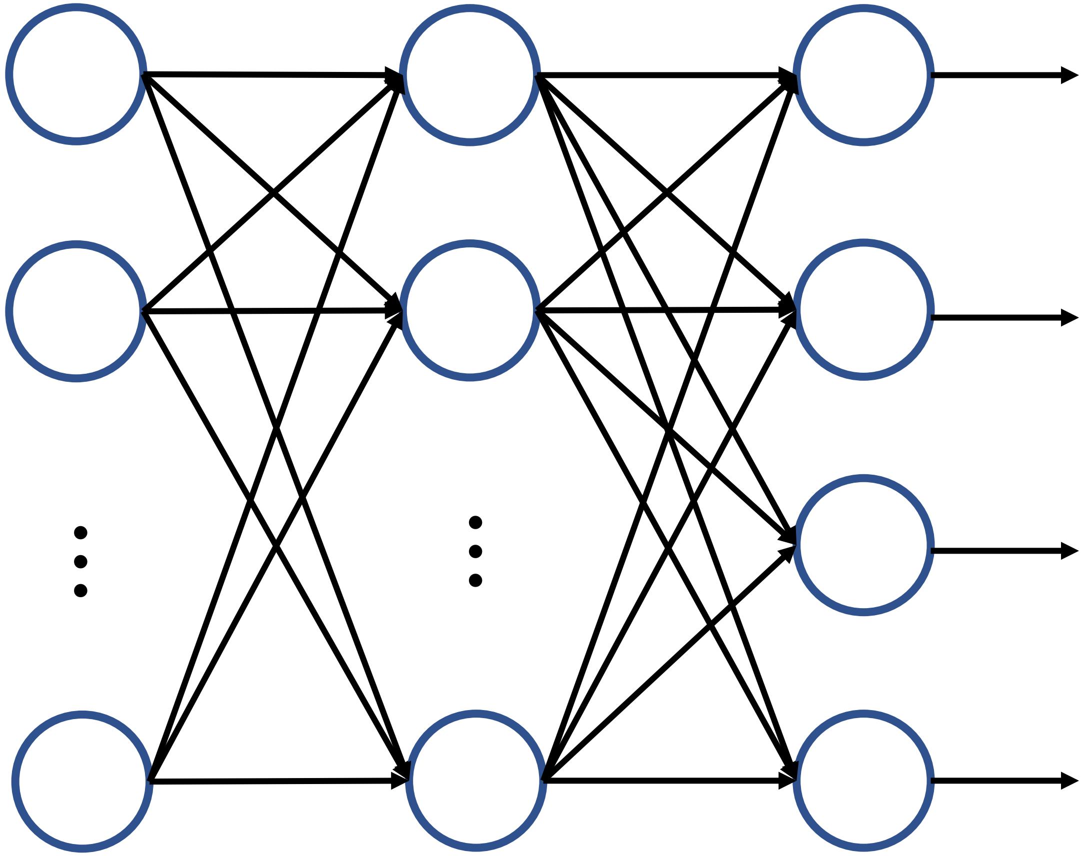
- Classes may overlap
  - Sample can be in more than one class
  - Sample may be in none of the classes



https://scikit-learn.org/stable/auto_examples/linear_model/plot_logistic_multinomial.html

Brad Quinton, Scott Chin

Activation on Output Layer



Layer 0
(Input Layer)

Layer 1
(Hidden Layer)

Layer 2
(Output Layer)

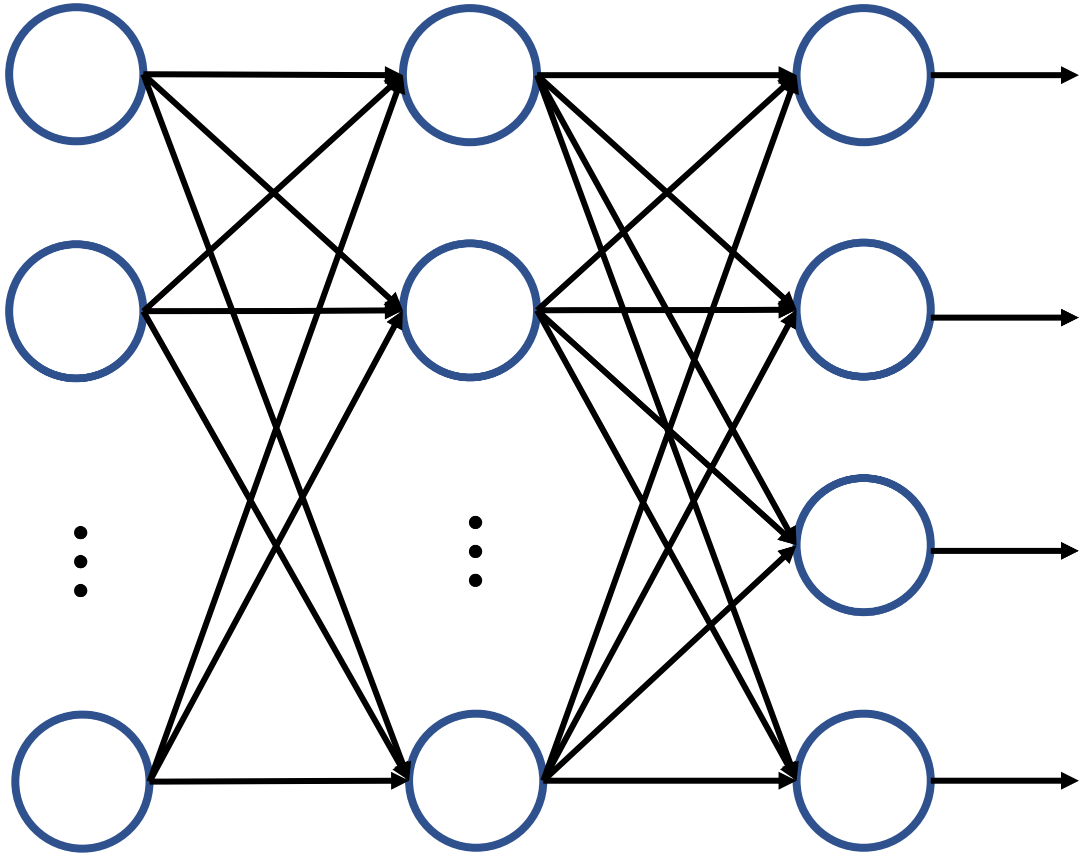
X

$Z^{[1]} = W^{[1]}X + B^{[1]}$
$A^{[1]} = g(Z^{[1]})$

$Z^{[2]} = W^{[2]}A^{[1]} + B^{[2]}$
$A^{[2]} = g(Z^{[2]})$

Brad Quinton, Scott Chin

Activation on Output Layer



Layer 0 (Input Layer)

Layer 1 (Hidden Layer)

Layer 2 (Output Layer)

X

$$Z^{[1]} = W^{[1]}X + B^{[1]}$$
$$A^{[1]} = g\left(Z^{[1]}\right)$$

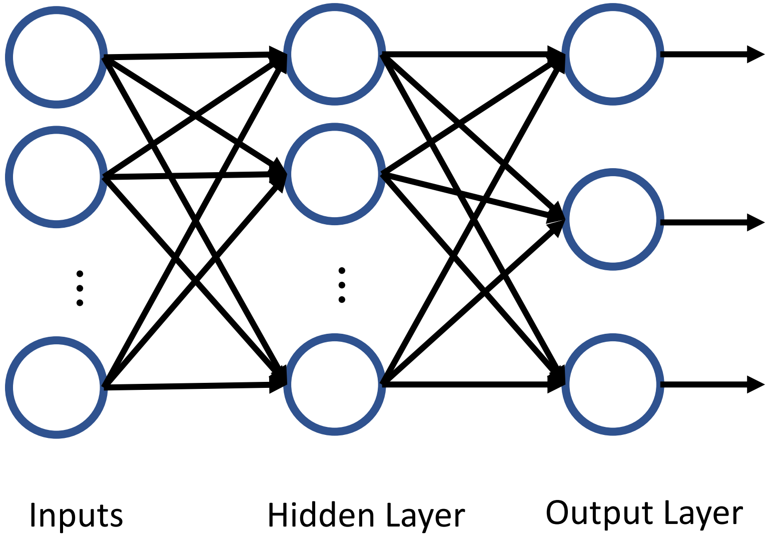
$$Z^{[2]} = W^{[2]}A^{[1]} + B^{[2]}$$
$$A^{[2]} = g\left(Z^{[2]}\right)$$

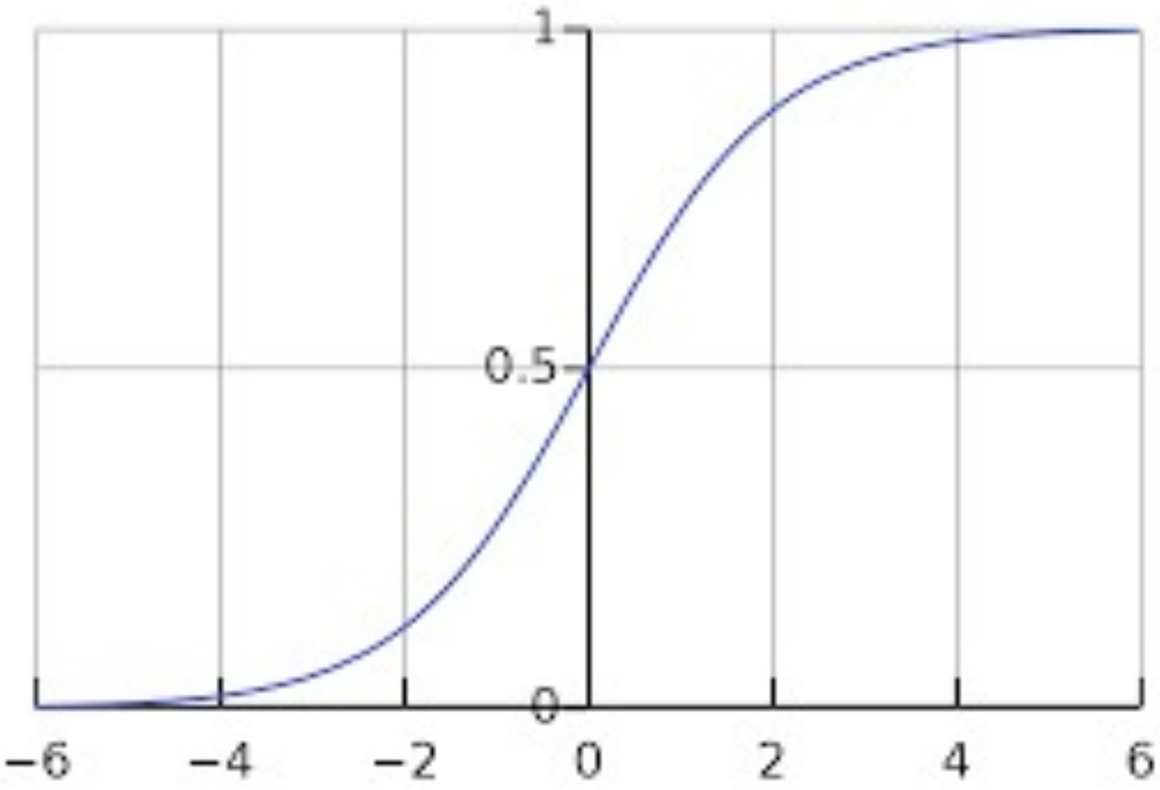
What activation function to use on output layer?

Brad Quinton, Scott Chin

Activation on Output Layer

- Can we still use Sigmoid?
- Kind of…

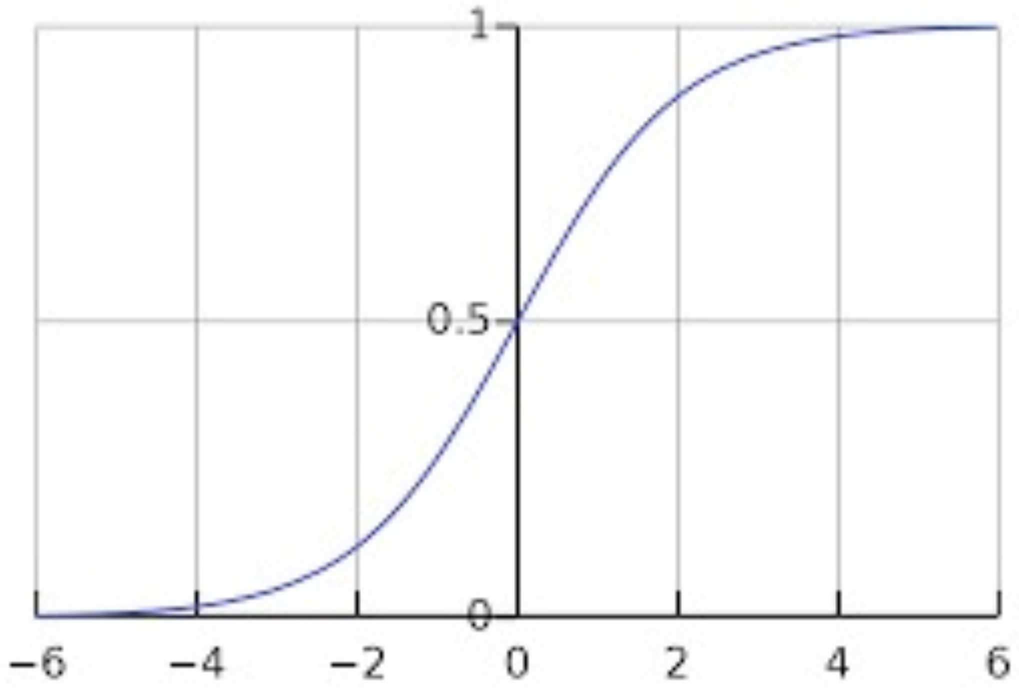




Brad Quinton, Scott Chin

Recall Sigmoid Activation Function

- Sigmoid produces an output **between** 0 and 1.0
- Can interpret this as a probability.
- For example
  - you have a Spam/Not Spam classifier.
  - Model prediction of 0.8 may be interpreted as 80% chance of email being spam.
  - Implicitly this means a 20% of the other class (not spam)



Brad Quinton, Scott Chin

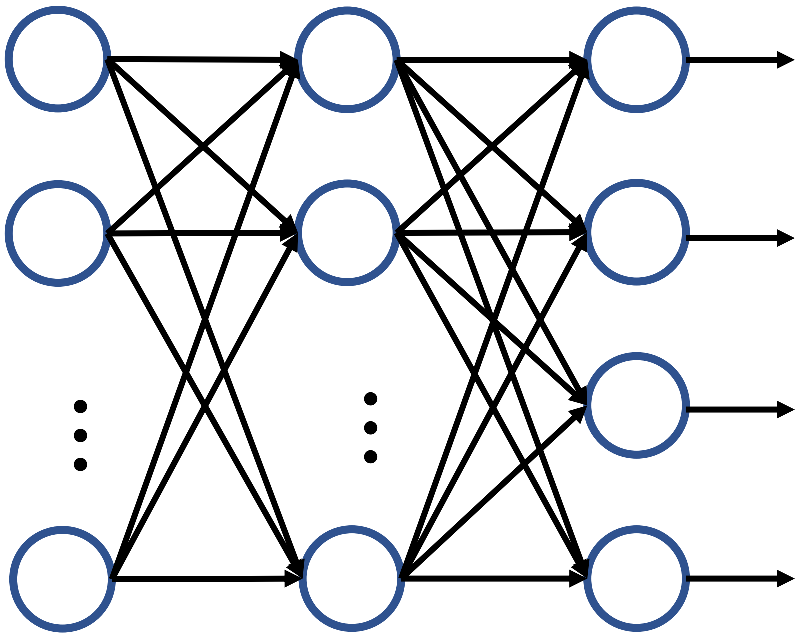
Softmax Activation Function

Brad Quinton, Scott Chin

Softmax Intuition

- Softmax activation normalizes the outputs such that each output node continues to produce a value between 0 and 1.0, and **also** sum to 1.0.

Example 4 output classes



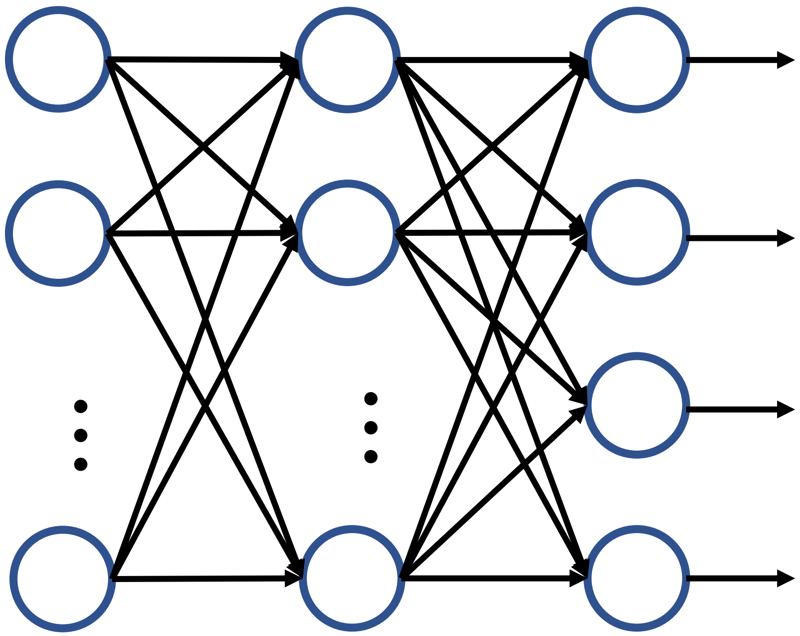
$$\begin{bmatrix} 0.79 \\ 0.07 \\ 0.11 \\ 0.03 \end{bmatrix}$$

$$Z^{[2]} = W^{[2]}A^{[1]} + B^{[2]}$$
$$A^{[2]} = g\left(Z^{[2]}\right)$$

Softmax Intuition - Probabilities

- We can interpret this as a set of prediction probabilities for each class

Example 4 output classes



$$\begin{bmatrix} 0.79 \\ 0.07 \\ 0.11 \\ 0.03 \end{bmatrix}$$

$$Z^{[2]} = W^{[2]}A^{[1]} + B^{[2]}$$
$$A^{[2]} = g\left(Z^{[2]}\right)$$

Interpret as probabilities

$$\begin{bmatrix} p(class_0|x) \\ p(class_1|x) \\ p(class_2|x) \\ p(class_3|x) \end{bmatrix}$$

Brad Quinton, Scott Chin

Softmax Definition

- Input is vector Z of length n_c
- Softmax produces a vector where each element is

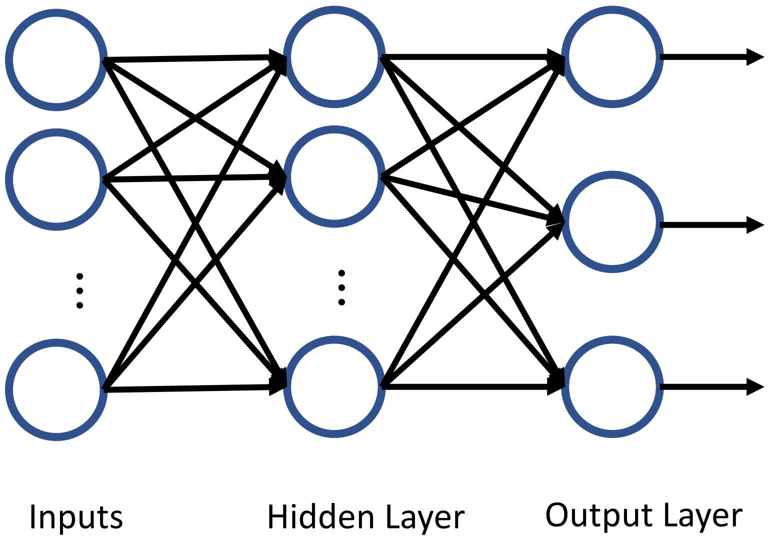
$$g_i(Z) = \frac{e^{z_i}}{\sum_{j=1}^{n_c} e^{z_j}}$$

- Each element is a value between 0 and 1.
- Sum of elements of the output vector is equal to 1

Brad Quinton, Scott Chin

Softmax Example

$$g_i(Z) = \frac{e^{z_i}}{\sum_{j=1}^{n_c} e^{z_j}} \qquad Z = \begin{bmatrix} -3.44 \\ 1.6 \\ 0.81 \\ 3.91 \end{bmatrix}$$



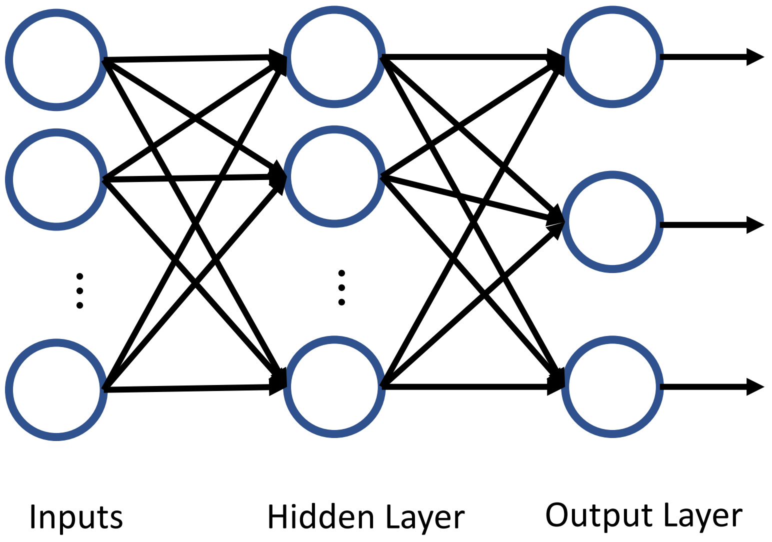
Brad Quinton, Scott Chin

Softmax Example

$$g_i(Z) = \frac{e^{z_i}}{\sum_{j=1}^{n_c} e^{z_j}}$$

$$Z = \begin{bmatrix} -3.44 \\ 1.6 \\ 0.81 \\ 3.91 \end{bmatrix}$$

$$numerator = \begin{bmatrix} e^{-3.44} \\ e^{1.6} \\ e^{0.81} \\ e^{3.91} \end{bmatrix} = \begin{bmatrix} 0.032 \\ 4.950 \\ 2.247 \\ 49.899 \end{bmatrix}$$



Brad Quinton, Scott Chin

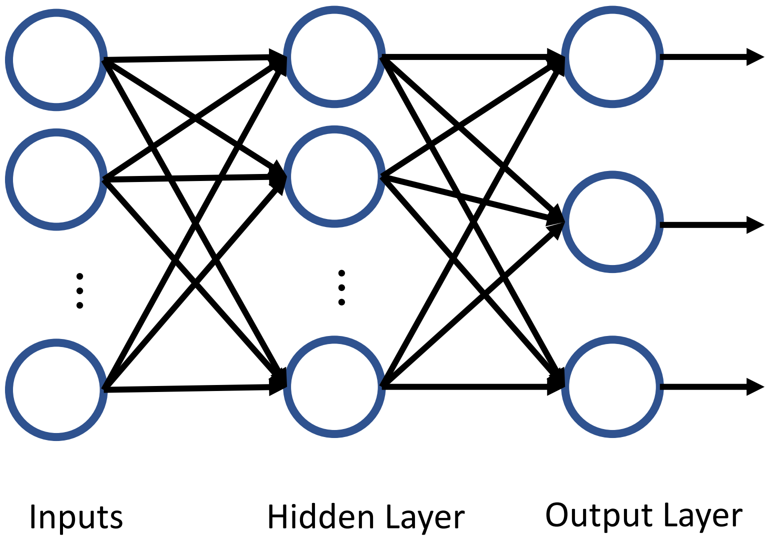
Softmax Example

$$g_i(Z) = \frac{e^{z_i}}{\sum_{j=1}^{n_c} e^{z_j}}$$

$$Z = \begin{bmatrix} -3.44 \\ 1.6 \\ 0.81 \\ 3.91 \end{bmatrix}$$

$$numerator = \begin{bmatrix} e^{-3.44} \\ e^{1.6} \\ e^{0.81} \\ e^{3.91} \end{bmatrix} = \begin{bmatrix} 0.032 \\ 4.950 \\ 2.247 \\ 49.899 \end{bmatrix}$$

$$denominator$$
$$= e^{-3.44} + e^{1.6} + e^{0.81} + e^{3.91}$$
$$= 0.032 + 4.950 + 2.247 + 49.899$$
$$= 57.128$$



Brad Quinton, Scott Chin

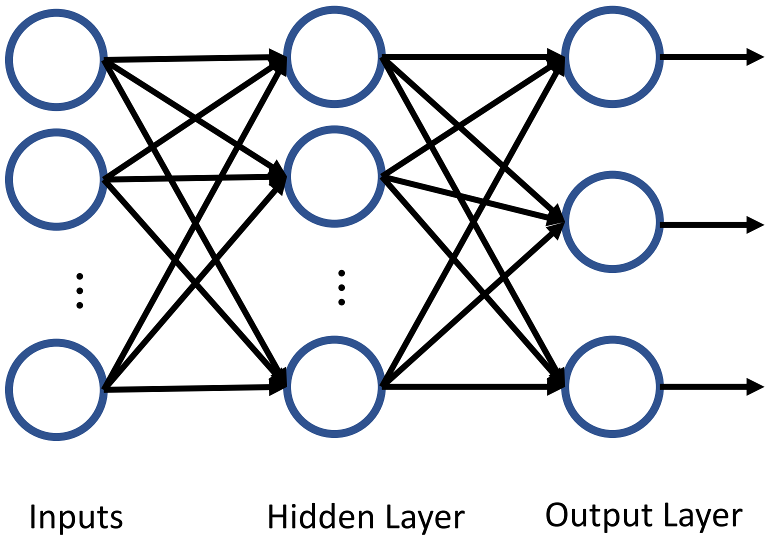
Softmax Example

$$g_i(Z) = \frac{e^{z_i}}{\sum_{j=1}^{n_c} e^{z_j}}$$

$$Z = \begin{bmatrix} -3.44 \\ 1.6 \\ 0.81 \\ 3.91 \end{bmatrix}$$

$$numerator = \begin{bmatrix} e^{-3.44} \\ e^{1.6} \\ e^{0.81} \\ e^{3.91} \end{bmatrix} = \begin{bmatrix} 0.032 \\ 4.950 \\ 2.247 \\ 49.899 \end{bmatrix}$$

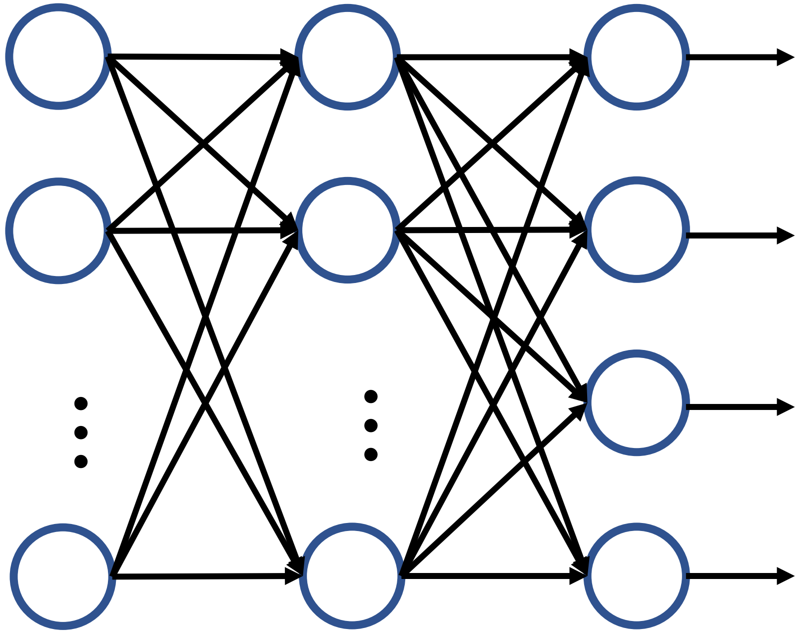
$$\begin{aligned} &denominator \\ &= e^{-3.44} + e^{1.6} + e^{0.81} + e^{3.91} \\ &= 0.032 + 4.950 + 2.247 + 49.899 \\ &= 57.128 \end{aligned}$$



$$softmax(Z) = \begin{bmatrix} 0.032 \\ 4.950 \\ 2.247 \\ 49.899 \end{bmatrix} * \frac{1}{57.128} = \begin{bmatrix} 0.0006 \\ 0.087 \\ 0.039 \\ 0.873 \end{bmatrix}$$

At Prediction Time

- Pick the class with highest probability



$$\begin{bmatrix} 0.0006 \\ 0.087 \\ 0.039 \\ 0.873 \end{bmatrix}$$

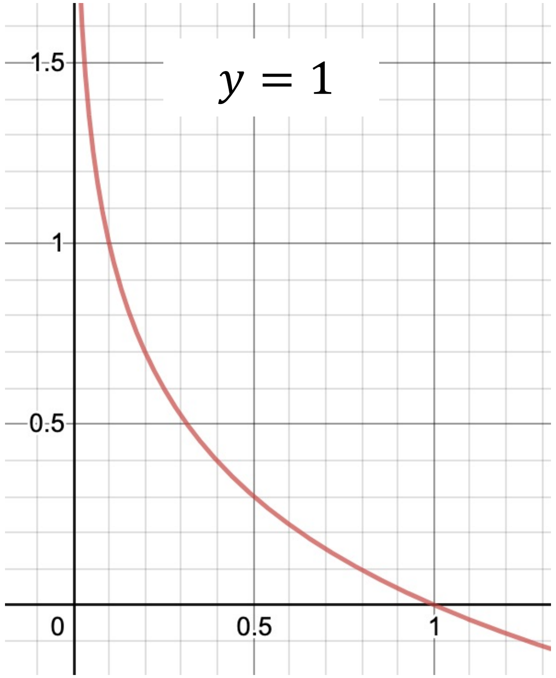
Brad Quinton, Scott Chin

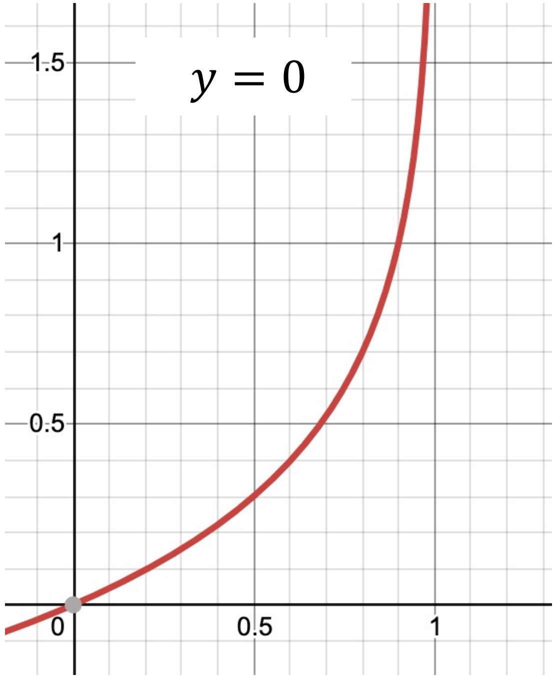
How to measure loss on softmax output?

- Recall the Binary Cross Entropy Loss:

$$L(\hat{y}, y) = -(y\log(\hat{y}) + (1-y)\log(1-\hat{y}))$$

$$L(\hat{y}, y) = \begin{cases} -\log(\hat{y}), & y = 1 \\ -\log(1-\hat{y}), & y = 0 \end{cases}$$





Brad Quinton, Scott Chin

How to measure loss on softmax output?

- Categorical Cross Entropy Loss (Sometimes called Softmax Loss) is a generalization of the Binary Cross Entropy Loss

$$L(\hat{y}, y) = -\sum_{j=1}^{n_C} y_j log\left(\hat{y}_j\right)$$

Brad Quinton, Scott Chin

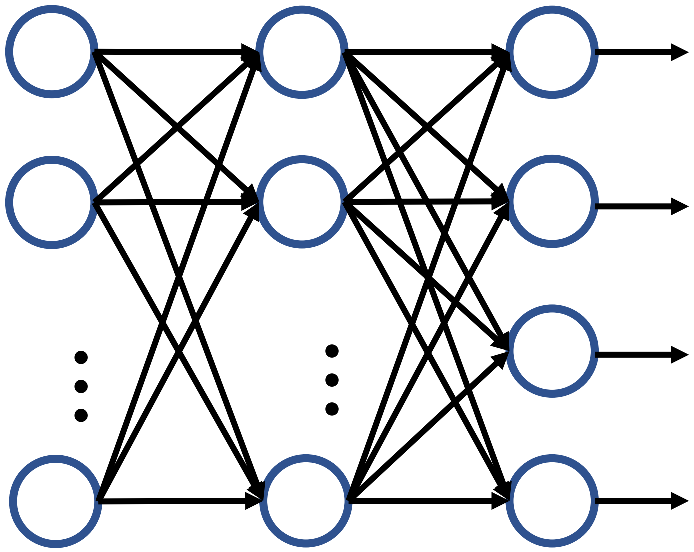
Cross-Entropy from Information Theory

- Cross Entropy quantifies the difference between two probability distributions over the same underlying set of events
  - A true distribution (the true labels)
  - An estimated distribution (the model's predicted label)

- Therefore, by minimizing Cross Entropy, we are trying to make the predicted output equal to the true output

Brad Quinton, Scott Chin

Categorical Cross Entropy Loss - Example



$$\hat{y} = \begin{array}{c} \begin{bmatrix} 0.0006 \\ 0.087 \\ 0.039 \\ 0.873 \end{bmatrix} \end{array} \begin{array}{c} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \hat{y}_4 \end{array} \qquad y = \begin{array}{c} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \end{array} \begin{array}{c} y_1 \\ y_2 \\ y_3 \\ y_4 \end{array}$$

$$L(\hat{y}, y) = -\sum_{j=1}^{n_c} y_j log(\hat{y}_j) = ???$$

Brad Quinton, Scott Chin

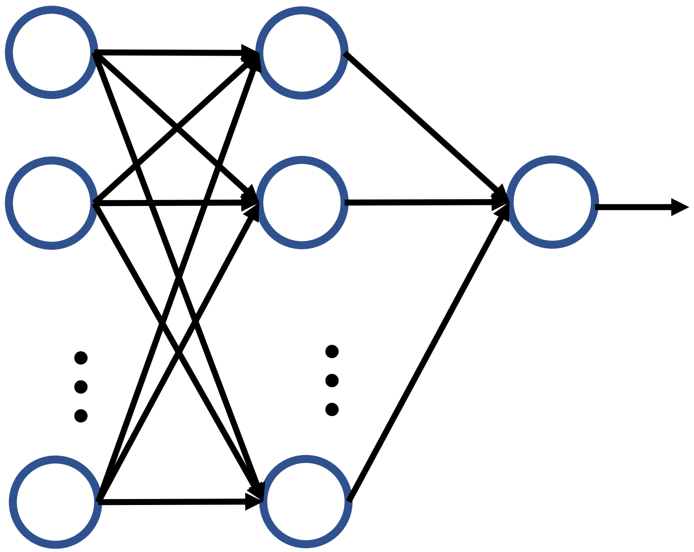
Softmax is a Generalizaton of Sigmoid

Brad Quinton, Scott Chin

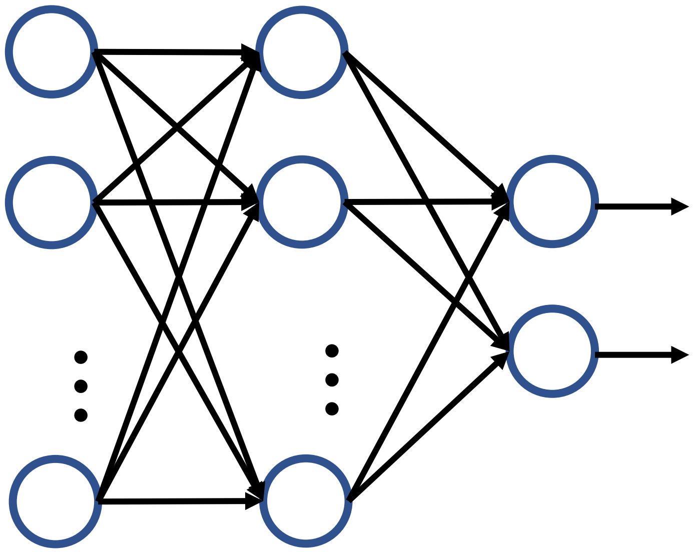
Softmax is a Generalization of Sigmoid

We will show that for the case of two classes

- Softmax function is equivalent to Sigmoid function
- Categorical Cross Entropy Loss is equivalent to Binary Cross Entropy Loss



Using Sigmoid



Using Softmax

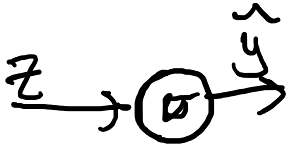
Brad Quinton, Scott Chin

Hand Written Notes: Softmax with 2 class is equivalent to Sigmoid

Brad Quinton, Scott Chin

SIGMOID :

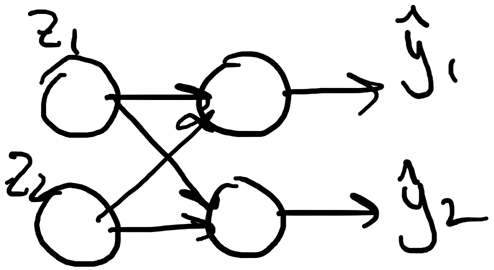
$$\sigma(z) = \frac{1}{1+e^{-x}} \Rightarrow p(\hat{y}=1 \mid z)$$



$$z = wx + b$$

$$p(\hat{y}=0 \mid z) = 1 - p(\hat{y}=1 \mid z) = 1 - \frac{1}{1+e^{-z}} = \frac{e^{-z}}{1+e^{-z}}$$

SOFTMAX FOR $n_c = 2$



$$\hat{y}_1 = \frac{e^{z_1}}{e^{z_1}+e^{z_2}} = \frac{e^{z_1}}{e^{z_1}+e^{z_2}} \cdot \frac{1/e^{z_1}}{1/e^{z_1}} = \frac{1}{1+e^{z_2-z_1}}$$

$$\hat{y}_2 = \frac{e^{z_2}}{e^{z_1}+e^{z_2}} = \frac{e^{z_2}}{e^{z_1}+e^{z_2}} \cdot \frac{1/e^{z_1}}{1/e^{z_1}} = \frac{e^{z_2-z_1}}{1+e^{z_2-z_1}}$$

$$z_1 = w_1x + b_1$$
$$z_2 = w_2x + b_2$$
$$z = wx + b$$

$$z = z_1 - z_2$$
$$wx + b = w_1x + b_1 - (w_2x + b_2)$$
$$= (w_1 - w_2)x + (b_1 - b_2)$$

THEN FOR SOME $w = w_1 - w_2$

$b = b_1 - b_2$

LET $z = z_1 - z_2$

THEN $\hat{y}_1 = \frac{1}{1+e^{-z}}$

$\hat{y}_2 = \frac{e^{-z}}{\ldots}$

Brad Quinton, Scott Chin

Hand Written Notes: Categorical Cross Entropy Loss is the generalization of Binary Cross Entropy Loss

BINARY CROSS ENTROPY LOSS

$$L_{BIN}(\hat{y}, y) = -\left(y \log \hat{y} + (1-y)\log(1-\hat{y})\right)$$

CATEGORICAL CROSS ENTROPY LOSS FOR $n_c = 2$ (TWO CLASSES)

$$L_{CAT}(\hat{y}, y) = -\sum_{j=1}^{n_c} y_j \log \hat{y}_j = -\left(y_1 \log \hat{y}_1 + y_2 \log \hat{y}_2\right)$$

RECALL ONE-HOT ENCODED y

$$\sum_{j=1}^{n_c} y_j = 1 \quad \therefore \quad y_2 = 1 - y_1$$

FOR SOFTMAX

$$\sum_{j=1}^{n_c} \hat{y}_j = 1 \quad \therefore \quad \hat{y}_2 = 1 - \hat{y}_1$$

$$\therefore \ L_{CAT}(\hat{y}, y) = -\left(y_1 \log \hat{y}_1 + (1-y_1)\log(1-\hat{y}_1)\right)$$

LET $\hat{y} = P(y=1 | x)$ AND $\hat{y}_1 = P(y_1 = 1 | x)$

Cost Function

- Nothing new here. Just like before we want to minimize the average loss across all training samples

$$J(W, B) = \frac{1}{m} \sum_{i=1}^{m} L\left(\hat{y}^{(i)}, y^{(i)}\right)$$

Back Propagation through Softmax Layer

Hand Written Notes: Backprop through softmax layer

$$L(\hat{y}, y) = \sum_{j=1}^{n} y_j \log \hat{y}_j = -y_1 \log \hat{y}_1 - y_2 \log \hat{y}_2 - y_3 \log \hat{y}_3$$

$$\frac{\partial L}{\partial z_1} = -y_1 \frac{\partial(\log \hat{y}_1)}{\partial z_1} - y_2 \frac{\partial(\log \hat{y}_2)}{\partial z_1} - y_3 \frac{\partial(\log \hat{y}_3)}{\partial z_1} = -y_1 \frac{\partial(\log \hat{y}_1)}{\partial \hat{y}_1} \cdot \frac{\partial \hat{y}_1}{\partial z_1} + -y_2 \frac{\partial(\log \hat{y}_2)}{\partial \hat{y}_2} \cdot \frac{\partial \hat{y}_2}{\partial z_1} + \ldots$$

$$= -y_1 \cdot \frac{1}{\hat{y}_1} \cdot \frac{\partial \hat{y}_1}{\partial z_1} + -y_2 \cdot \frac{1}{\hat{y}_2} \cdot \frac{\partial \hat{y}_2}{\partial z_1} + -y_3 \cdot \frac{1}{\hat{y}_3} \cdot \frac{\partial \hat{y}_3}{\partial z_1}$$

$$\hat{y}_1 = \frac{e^{z_1}}{e^{z_1}+e^{z_2}+e^{z_3}} \quad \hat{y}_2 = \frac{e^{z_2}}{e^{z_1}+e^{z_2}+e^{z_3}} \quad \hat{y}_3 = \frac{e^{z_3}}{e^{z_1}+e^{z_2}+e^{z_3}} \quad \text{Let } t = e^{z_1}+e^{z_2}+e^{z_3}$$

$$\frac{\partial \hat{y}_1}{\partial z_1} = \frac{e^{z_1} t - e^{z_1} e^{z_1}}{t^2} = \frac{e^{z_1}}{t} - \left(\frac{e^{z_1}}{t}\right)^2 = \hat{y}_1 - (\hat{y}_1)^2 = \hat{y}_1(1-\hat{y}_1)$$

$$\frac{\partial \hat{y}_2}{\partial z_1} = \frac{0 \cdot t - e^{z_2} e^{z_1}}{t^2} = -\frac{e^{z_2}}{t} \cdot \frac{e^{z_1}}{t} = -\hat{y}_2 \cdot \hat{y}_1 \quad \Bigg| \quad \frac{\partial \hat{y}_3}{\partial z_1} = \frac{0 \cdot t - e^{z_3} \cdot e^{z_1}}{t^2} = -\frac{e^{z_3}}{t} \cdot \frac{e^{z_1}}{t} = -\hat{y}_3 \cdot \hat{y}_1$$

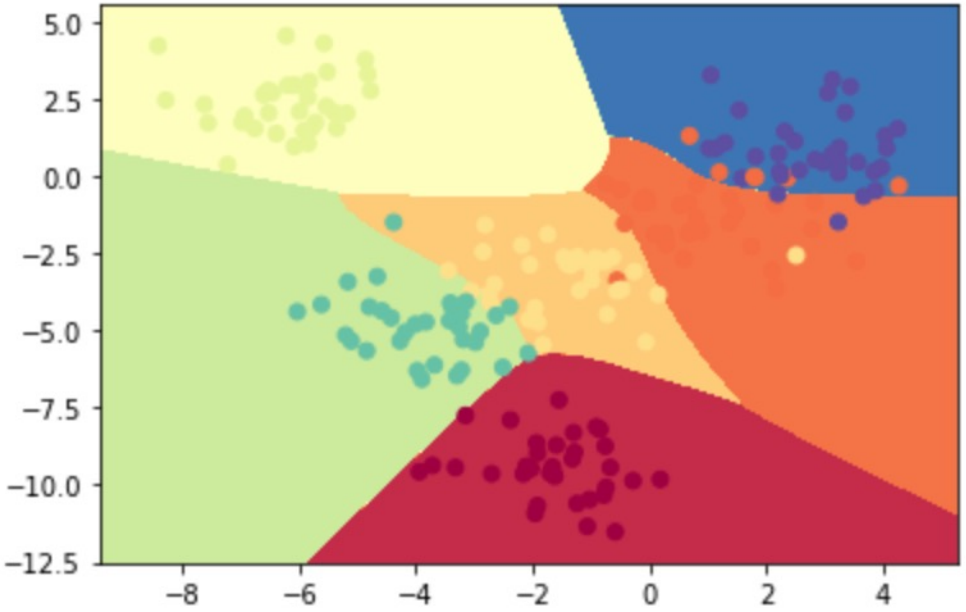
$$= -\frac{y_1}{\hat{y}_1} \cdot \hat{y}_1(1-\hat{y}_1) - \frac{y_2}{\hat{y}_2} \cdot (-\hat{y}_2 \cdot \hat{y}_1) - \frac{y_3}{\hat{y}_3} \cdot (-\hat{y}_3 \hat{y}_1) = -y_1 + y_1 \hat{y}_1 + y_2 \hat{y}_1 + y_3 \hat{y}_1$$

$$= -y_1 + \hat{y}_1 \underbrace{(y_1 + y_2 + y_3)}_{=1} = -y_1 + \hat{y}_1 = \boxed{\hat{y}_1 - y_1} \quad \frac{\partial L}{\partial z_2} = \hat{y}_2 - y_2 \quad \frac{\partial L}{\partial z_3} = \hat{y}_3 - y_2$$

Summary of Multiclass Classification (Single Neural Network with Multiple Outputs)

- One output node for each class
- Use Softmax activation function on final layer of output nodes
- Minimize the Categorical Cross-Entropy Loss
- Train on one-hot encoded label data
- Cannot be used for Multi-Label Classification

Example: Decision Regions for 2 Features, 6 classes



Multilabel Classification

- Cannot use softmax
- Use separate classifiers or use Sigmoids on outputs
- Labels cannot be one-hot encoded vectors

Brad Quinton, Scott Chin

Learning Objectives

- The Multiclass Classification Problem
- How to encode the output for a Neural Network
- Common approaches to Multiclass Classification
- Softmax Activation Function
- Categorical Cross-Entropy Loss
- Back Propagation through Softmax Layer