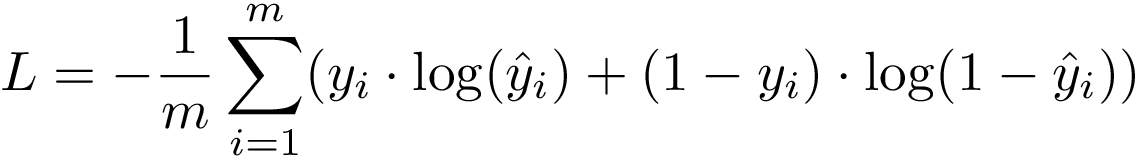
NG Challenges 1

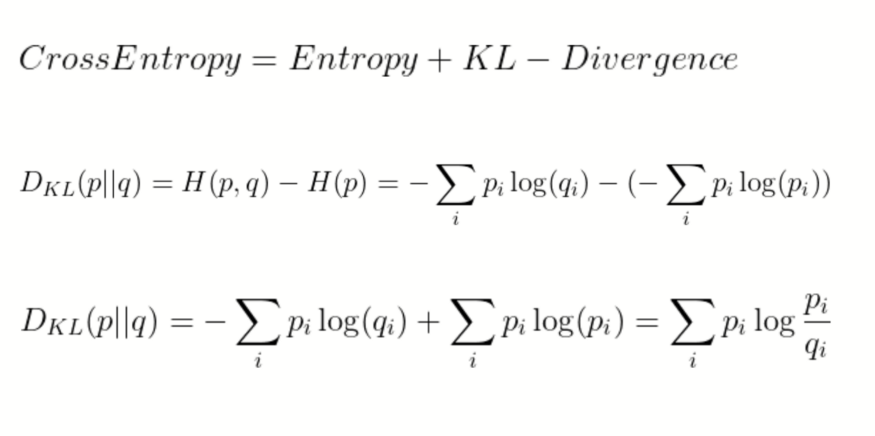
* Load MNIST dataset with Keras dataset command
* Split the dataset into 70% training dataset and 30% testing dataset.
* Imitate the network LeNet, you may load it with Keras or build it manually.
* Conduct initializer experiment. Build network with Glorot, He, and Random Normal initializer. Train these networks for a short time (10-20 epoch), and choose the initializer with the best result for our next stage of training.
* Activation Functions: Modify the activation function on the convolution layers, train one network with Sigmoid and another network with ReLU. You may use any initializer.
  + Optimizers: Train the network with ReLU and Glorot initializer with different initializers; SGD with Nesterov Momentum, Adam, and RMSProp. You also need to do hyperparameter tuning for each of the optimizers, with minimum 3 different hyperparameter setting for each of the optimizers.
* Find the most optimal hyperparameter settings.
* Observe and analyze the result of your experiment. You can observe the result via its learning curve (error vs epoch), or further via the prediction of the network. Your analysis may contain points such as:
  + Effect of using different activation functions on the learning curve.
  + Effect of using different optimizers on learning curve.
  + Effect of hyperparameter tuning on the optimizers.
  + Which are the most optimal settings for training a network for MNIST problem, and assess why it produces the best result.

NG Challenges 2.

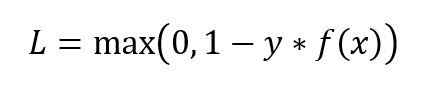
* Build 2 50x2 dimension Tensor with random binary values and their probability of 0 and 1 is 0.5.
* Use the first generated tensor as target, and the other vector as prediction. We will use these vectors to learn custom losses and metrics.
* Build the following losses used for classification with custom functions:
  + Cross entropy loss



* + KL-Divergence



* + Hinge loss.



* Use the loss to compare the two tensors that you have built previously. Compare the result of your custom function with the default loss function provided by Keras.
* Build F1 score calculation with custom metrics.

Milestone

We have learned how to build a neural network, now we will use it in an ML Pipeline.

This milestone project can be splitted into two. First part of our milestone project is image recognition phase, and the second one is numerical predictive modelling.

Here are the steps:

1. Download the Pokemon image DB here:

<https://www.kaggle.com/vishalsubbiah/pokemon-images-and-types>

1. Encode the Pokemon primary Type, and we will use them as label.
2. Use the Pokemon image, and try to determine Pokemon Type from its appearance.

Next step is combined recognition. Here, we will combine the output of our neural network with an existing DB. Here are the steps:

1. Download the Pokemon stats here:

https://www.kaggle.com/abcsds/pokemon

1. Again, encode the Pokemon primary type with the same mapping as we used on image recognition problem.
2. Now, combine the prediction output from your Neural Network with the numerical stat that we have from the data, then build a prediction model that take all the data to predict the Pokemon type. You can join the data by the Pokemon name.
3. Remember, we are now faced with numerical data, so preprocessing such as normalization etc. is critical. Hint: One of the preprocessing you can done is relative scaling, i.e. changing stat from absolute value to relative percentage to total of the stat.
4. Try to experiments with many model to obtain the optimal model. You can also try using neural network as the model.
5. To improve the model, it is also beneficial to understand how Pokemon stat works. For example, knowing how Pokemon level system, legendary Pokemon type, etc. This is also what happen in real life where you analyse data, that it is always good to know the business or underlying mechanism on how the data is produced in nature.