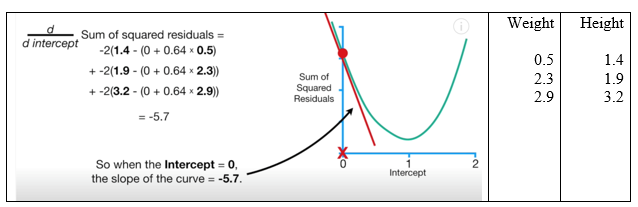
**COMP4948 Final Study Guide**

**On final:**

* Explain how the loss function is used in a stochastic gradient algorithm.
  + LESSON 1, LAB 1
  + Loss function is a method of evaluating how well an algorithm models a dataset
  + It is lower when the model makes accurate predictions
    - Is really just the sum of the squared residuals SSR
  + It describes how well a model will perform given the current set of weights and biases
  + SGD fits linear classifiers and regressors under convex loss functions such as logistic regression
  + Stochastic means random, SGD randomly picks one data point from the whole data set at each iteration and reduces computations compared to regular gradient descent
    - Regular gradient descent minimizes SSR
* Calculate step size in stochastic gradient descent.
  + Step size = slope \* learning rate
* Know the steps that are performed by a neural network.
  + LESSON 2
  + A neural network has an input layer, hidden layer, and output layer
  + Attributes are fed into the input layer 🡪 the hidden layer then updates or initializes weights and biases 🡪 passes weighted inputs and biases to the activation function 🡪 the output layer calculates loss, updates weights, then feeds the output back to the hidden layer to back propagate losses (step is repeated)
* Answer questions about when to use sparse\_categorical\_crossentropy, binary\_crossentropy and categorical crossentropy.
  + Binary cross entropy is used for a single output of 0 or 1
  + Sparse categorical entropy is used for categories expressed as a series of non-binary integers
  + Categorical cross entropy is helpful when output categories are hot-encoded
* Identify differences between MLP, CNN and LSTM architectures. Explain the strengths and advantages of each.
  + MLP are classical neural networks, comprised of one or more layers of neurons
    - Data fed into input layer, then hidden layers provide abstractions, and predictions are made in the visible layer/output layer
    - Suitable for classification predictions where inputs are assigned classes/labels and regression predictions where valued quantities are predicted given a set of inputs
    - Ex. Image data, text data, time-series data
  + Convolutional Networks (CNNS) were designed to map image data to an output variable
    - Are the go-to method for any image data prediction problems
    - Works well with data that has a spatial relationship ie. Images
    - Input 🡪 convolution 🡪 pooling 🡪 flattening 🡪 classification 🡪 output
  + Long short-term memory (network) is a type of RNN capapble of learning order dependence in sequence prediction problems
    - Have connections which have loops, adding feedback and memory to networks over time, allowing them to learn and generalize across sequences of inputs rather than individual patterns
    - LSTMS can be used for language translation or captioning of images and videos
    - Can be used for sentiment classification, speech captioning, machine translation, and video classification
* Answer questions about why and when ReLU is used.
* LESSON 4
  + ReLU is an activation function that was developed after sigmoid and tanh (which led to vanishing gradients which caused failures in weight updates during back propagation causing neurons to stop learning)
    - Activation functions are responsible for transforming the summed weight input from the node into the activation of the node or output for that input
  + The Rectified Linear Unit helps avoid vanishing gradients, the derivative of the activation function for positive values of X solves to 1 so gradients can back propagate
    - Neurons with negative values die, but enough will survive
  + A model using ReLU is easier to train and achieves better performance and replaces sigmoid and tanh functions in networks with many layers due to vanishing gradient
  + Default activation when developing MLP and CNN
* Be able to read and interpret loss and accuracy curves.
  + LAB 3
  + Loss explosion
    - NaNs in input data, division by 0 or logarithm of negative numbers
    - Exploding gradient due to anomalous data
  + Test curve goes above train curve and curls up
    - Model is overfitting to the training data, reduce capacity
      * Indicates the model has learned training data too well, including noise and random fluctuations
      * Problem is more specialized model becomes, less it is able to generalize new data resulting in increase in generalization error
    - Add regularization and check that the training/test splits are statistically equivalent
  + Train curve stays above validation and curves down
    - Model is underfitting and is capable of learning further and improving
    - Indicates training was halted prematurely
  + Good fit will go to 0 and train and validation curves stay together
* Define and questions about vanishing and exploding gradients, their symptoms, consequences and how to address them.
  + LAB 5
  + Exploding gradients occur when overflowing gradients are back-propagated
    - Situation happens mostly with LSTMs where the context from past gradients are included with the backpropagation
    - Happen due to improperly scaled inputs to the network, poorly chosen learning rates that allow for large weight updates, and loss functions which allow calculation of large error values
  + As more layers using activation functions are added to neural networks, the gradients of the loss function approaches 0 making the network hard to train
    - Occurs because certain activation functions like sigmoid squishes large input into a smaller input space between 0 and 1, therefore a large change in input will cause a small change in output 🡪 small derivative approaching 0
    - In networks with many layers this can make it impossible to train the model
    - Simple solution is to try different activation functions like ReLU
* Define and answer questions about why, when and how to adjust the following items:
  + learning rate, decay, kernel initializers, epochs, batch size, scaling, momentum, number of nodes, number of layers, gradient clipping, norm scaling.
    - Learning rate
      * Higher rates lead to volatile but quicker learning at the risk of overshooting the optimum
      * Lower rates take longer but offer smoother trends for loss and accuracy
    - Momentum (decaying weighted average of past gradients)
      * It helps smooth weight adjustments and overcome barriers to weight adjustments like steep curves/flat regions
      * Higher momentum may lead to weight changes that are not sensitive enough to changes in solution plane
    - Kernel Initializers
      * Use distribution to optimize initial model weights
      * Different algorithms for initializers can help prevent vanishing and exploding gradient which destroy a neuron’s ability to back-propagate data
    - Number of nodes (width)
      * Higher width increases capability and efficiency
      * Most learning problems can be solved simply by making it wide enough but has diminishing returns
    - Number of layers
      * Deeper networks are more efficient than single layer networks
      * Optimized number of layers should be chosen with same number of nodes that was optimized for a single layer
      * More layers now possible thanks to ReLU eliminating vanishing gradient problem
    - Batch size
      * range between 1 and size of training set
      * small batches provide better generalization to prevent overfitting but take longer to train and produce more volatile results
      * a large batch size is faster but can lead to overfitting
    - Decay rate
      * decay rate can help gradually reduce the learning rate by including a decay attribute
      * can help produce better model results by increasing accuracy and decreasing loss
    - Gradient clipping (tackles exploding gradients)
      * If the gradient is too large, clipping rescales it to keep it small
      * Ensures reasonable behavior of gradient descent in RNNS even if loss landscape is irregular
* Know how to configure a basic ANN input layer and output layer based on expected inputs and outputs.
  + LAB 2 LESSON 2
* Calculate the Gini impurity index for two nodes where each node indicates two possible outcomes. Select the optimal feature for the tree based on the Gini index.
  + LAB 6
* Be able to answer questions about the parameters of a random forest classifier. You do not need to know all of the different options for max\_features (int is fine for 'max\_features')
  + LESSON 6
* n\_estimators = number of trees in the forest.
* max\_features = max number of features considered for splitting a node. Settings include:
  + int, then consider max\_features features at each split.
  + float, then max\_features is a fraction and int(max\_features \* n\_features) features are considered at each split.
  + “auto”, then max\_features=sqrt(n\_features).
  + “sqrt”, then max\_features=sqrt(n\_features) (same as “auto”).
  + “log2”, then max\_features=log2(n\_features).
  + None, then max\_features=n\_features.
* max\_depth = max number of levels in each decision tree.
* min\_samples\_split = min number of data points placed in a node before the node is split.
* min\_samples\_leaf = min number of data points allowed in a leaf node.
* bootstrap = method for sampling data points (with or without replacement).
* Identify main differences between bagging, boosting and stacking.
  + LESSON 8
  + Are ensemble methods where we combine weak learners
  + Bagging
    - Averaging the predictions from weak learners to generate more stable results with less variance
  + Boosting
    - Boosting models are added to an ensemble sequentially instead of at once, each loop a new model is created and the new base-learner is trained while using errors of previous learners
      * Improve accuracy through training sequence of weak models compensating for weakness of predecessors
    - While bagging mainly reduces variance, boosting fits sequentially multiple weak learners in an adaptive way
      * Each model focuses its efforts on the most difficult observations
    - Boosting reduces bias and so should be used for models with low variance but high bias
  + Stacking
    - considers heterogenous weak learners whereas the others consider homogeneous weak learners
    - satcking can be used to improve accuracy
* Describe and identify how weighting of loss is used in boosting.

**Coding**

* Questions will be like the examples and exercises that are presented for the following topics:
* Early stopping and model save.
* Stacking or ensemble neural networks.
* Stacking or ensemble machine learning.