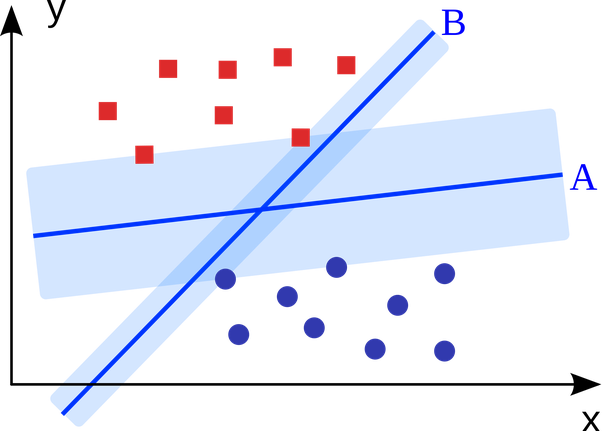
Name: David Josephs /10

1. **Support Vector Machines**
   1. What is the big idea behind Support Vector Machines? [0.5)

Support vector machines are used to clarify data. It works by finding a hyperplane with the maximum distance between the nearest data points.

* 1. As shown below, which of A and B is more likely to be found by SVM? Why? [0.5)

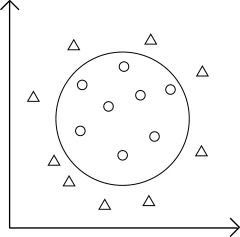


A. Because it separates the two groups according to the largest margin.

* 1. What is a support vector? Circle them in the figure above. [1]

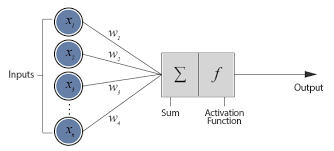
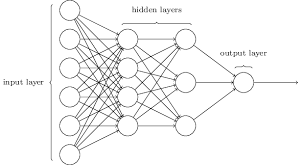
Support vectors are the samples on the margin.

* 1. What can be done to the data instances below such that SVM can be used to separate the triangles and the circles? [1]



Using a kernel trick such as transforming the data with z = sqrt(x^2+y^2) then plotting the data according to the new dimension z and x.

1. **Neural Networks**
   1. Which of (A) and (B) below is a perceptron? Write down the mathematical functions that it computes. [1]



(A) (B)

B.

Output = f(Sum(xi + Wi) for i = 1 to N)

* 1. What is an activation function? Provide two examples of activation functions. [1]

An activation function is a function that takes inputs and then usually outputs either a 1 or a 0.

Hyperbolic tangent function

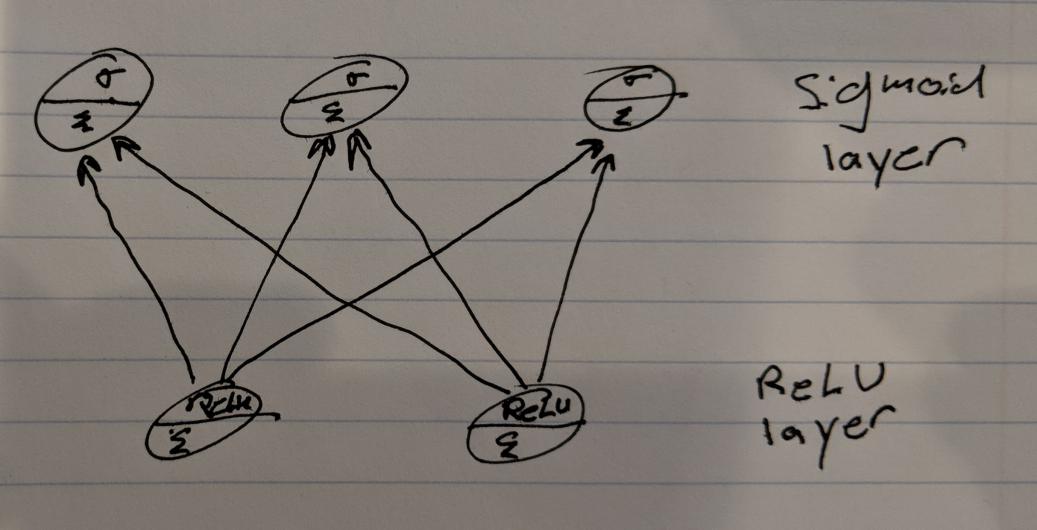
tanh(z) = 2sigma(2z)-1

The ReLU function

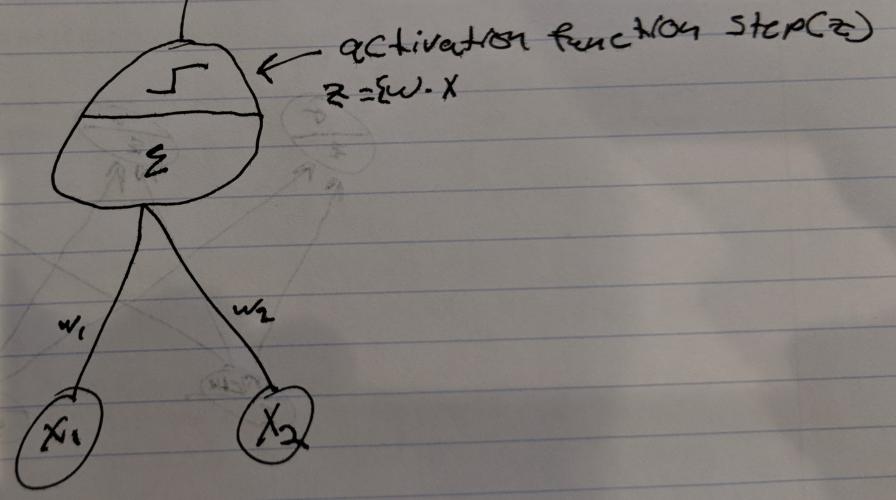
ReLU(z) = max(0,z)

* 1. Suppose there are two consecutive layers in a deep neural network where the upper layer has 3 neurons using sigmoid, and the bottom layer has 2 neurons using ReLU. Draw a diagram for the two layers. How many weights will there be between the two layers? [1]

There are 6 different weights between the layers.



1. **TensorFlow**  
   The code below implements an OR logic operator, defined as follows:   
   0 OR 1 = 1, 1 OR 0 = 1, 1 OR 1 = 1, 0 OR 0 = 0.   
   Answer questions about this code.
   1. Draw the network that this code implements. Mark the weights, sum, and activation function on your drawing. [1]



* 1. Explain tf.placeholder and tf.Variable and how they are different. [1]

tf.placeholder is for training values and is used to feed values into the model. tf.Variable will be changed according to the values in tf.placeholder.

tf.Variable holds the weights and biases for the Neural Network

* 1. What kind of error does it minimize? [1]

The error that the values are not true integers 1 and 0.

* 1. Explain how you would go about adding [0.99, 0.99] as a training instance. [1]

I would add [0.99,0.99] to the end of X\_batch and then add [1] to the end of Y\_batch

import numpy as np

import tensorflow as tf

n\_inputs = 2

n\_outputs = 1

X = tf.placeholder(tf.float32, shape=(None, n\_inputs), name="X")

y = tf.placeholder(tf.float32, shape=(None, n\_outputs), name="y")

# Construct model

W = tf.Variable(tf.zeros([n\_inputs,n\_outputs]))

b = tf.Variable(tf.zeros([n\_outputs]), name="bias")

Z = tf.matmul(X, W) + b

pred = tf.nn.sigmoid(Z)

# Minimize error using cross entropy

loss = tf.reduce\_mean(tf.reduce\_sum(tf.nn.l2\_loss(y-pred)))

learning\_rate = 0.01

optimizer = tf.train.GradientDescentOptimizer(learning\_rate)

training\_op = optimizer.minimize(loss)

init = tf.global\_variables\_initializer()

n\_epochs = 10000

with tf.Session() as sess:

X\_batch=[[1,0],[1,0.1],[1,0.02],[1,0.95],[1,1.01],[1,0.99],[1,1],[0,0],[0.1, 0.1]]

y\_batch = [[1], [1], [1], [1], [1],[1], [1], [0], [0]]

init.run()

for epoch in range(n\_epochs):

loss\_val, \_=sess.run([loss, training\_op], feed\_dict={X: X\_batch, y: y\_batch})

print("loss: ", loss\_val)

X\_new = [[1, 0.01], [1, 0.9], [1, 1], [1, 0.99], [0, 0.01], [0.01, 0.01]]

y\_pred = pred.eval(feed\_dict={X: X\_new})

y\_actuals = [1, 1, 1, 1, 0, 0]

print("Predicted classes:", np.round(y\_pred))

print("Actual classes: ", y\_actuals)