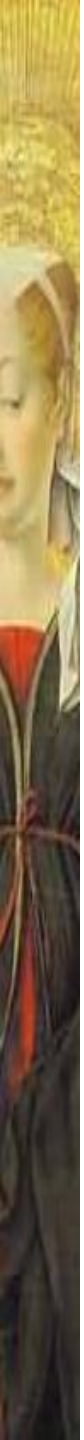
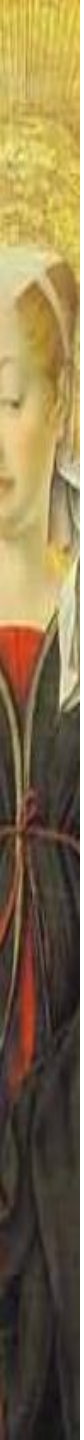


Topics in Machine Learning

CS212 Topics in Computing 2
Second Half
Lecture 5

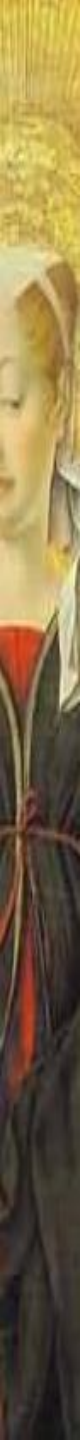
Term 2 2019

Dr. Dmitri Roussinov
dmitri.roussinov@strath.ac.uk



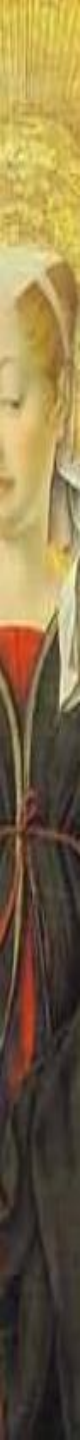
Test Next Week

- Closed book
- During the labs
- So we will not have a lecture
- Sample posted



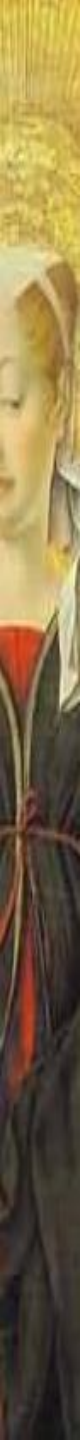
Lab 3 Tips

- Implementing a bias



Learned So far

- How a perceptron can make predictions
- What types of data it can learn
  - E.g. XOR vs AND
- Actual learning algorithm (binary)
  - Tested it on pictures of hand-written digits
  - Subsets only {0,1,2,3,4} vs. {5,6,7,8,9}
- Today: training for any number of classes
  - So we can finally recognize any digit
  - And not only digits!



Review: Perceptron and its Training Algorithm

Prediction:

a = 0

For i from 1 to length(W) do:

a += W[i] * X[i]

Y = 1 if a >= 0, 0 otherwise

Training:

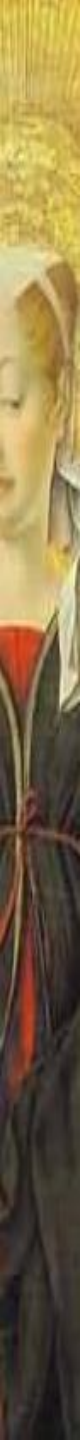
While error exists:

For all training examples do:

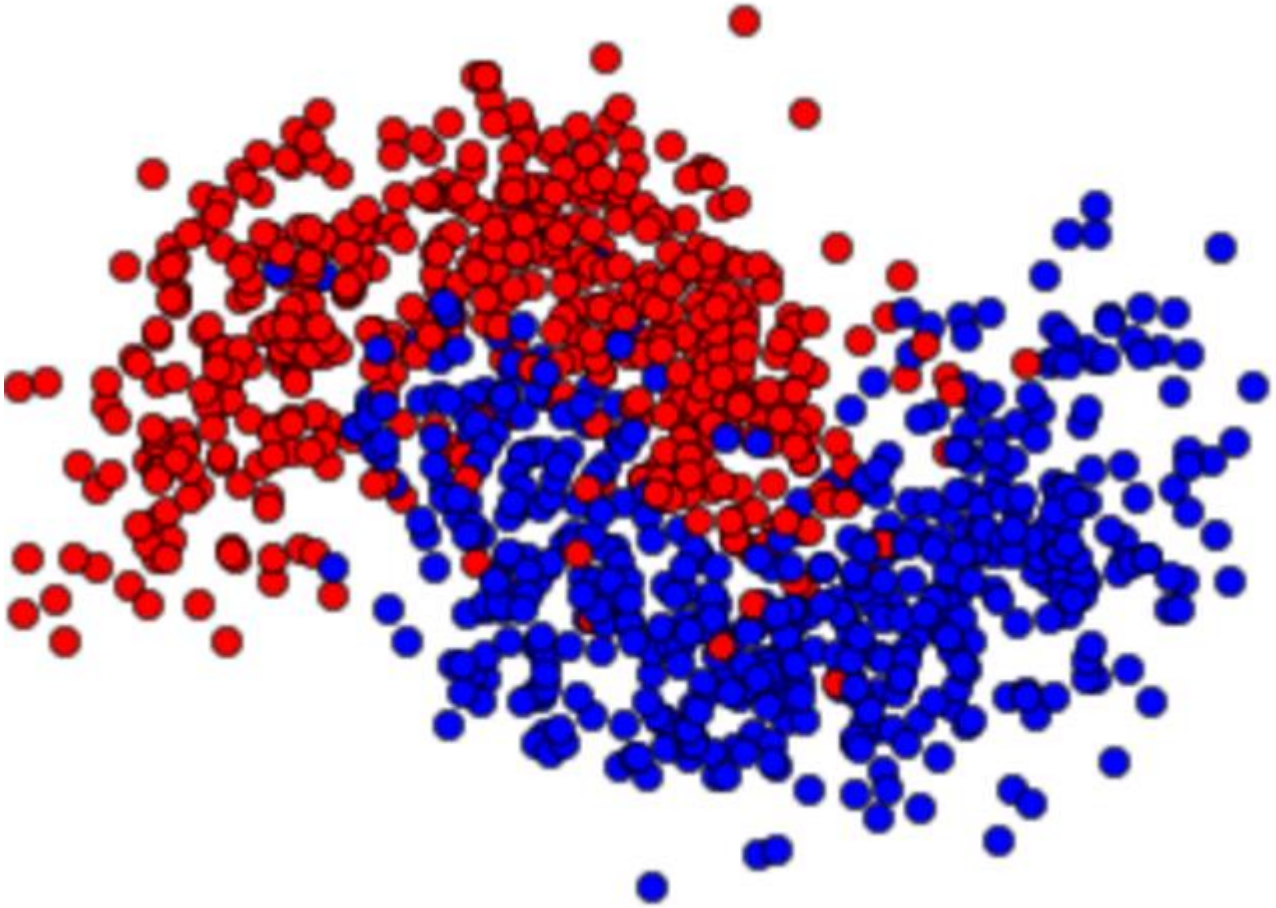
error = label - prediction

For i from 1 to length(W) do:

W[i] += learning_rate * error * X[i]

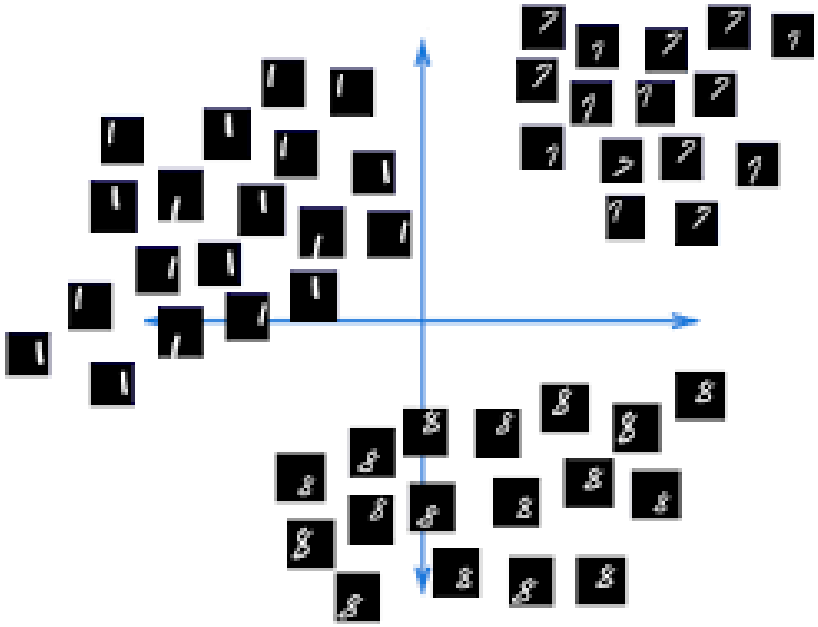


Review: Binary Classification

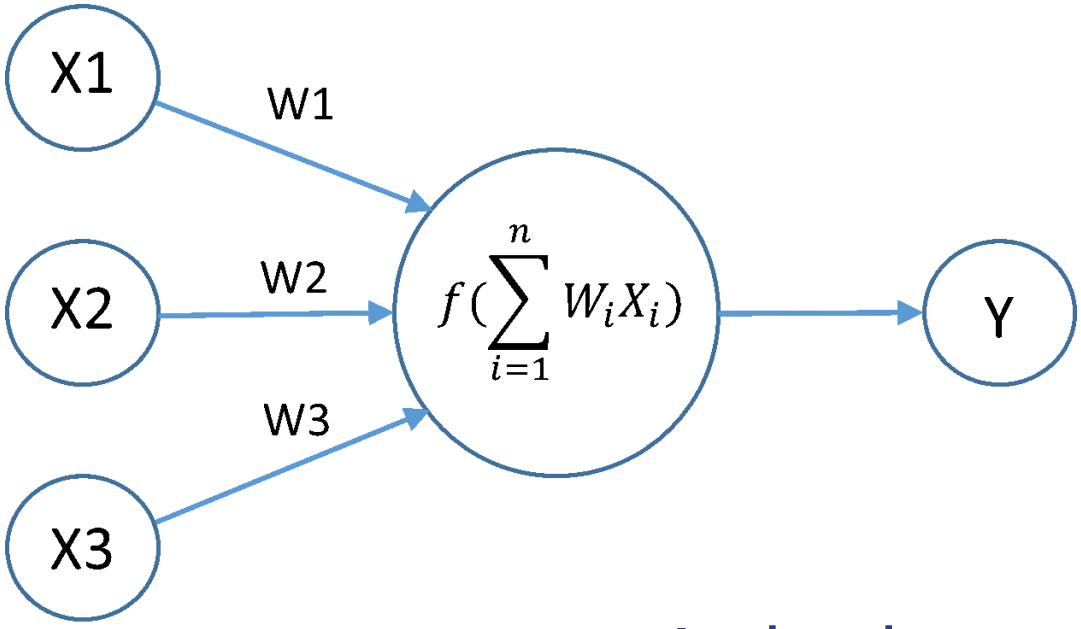


Multi-class problem

- So far we looked at binary predictions (classifications)
- Many real problems are mult class
- Including our MNIST digit recognition
- So, what do we need to change?

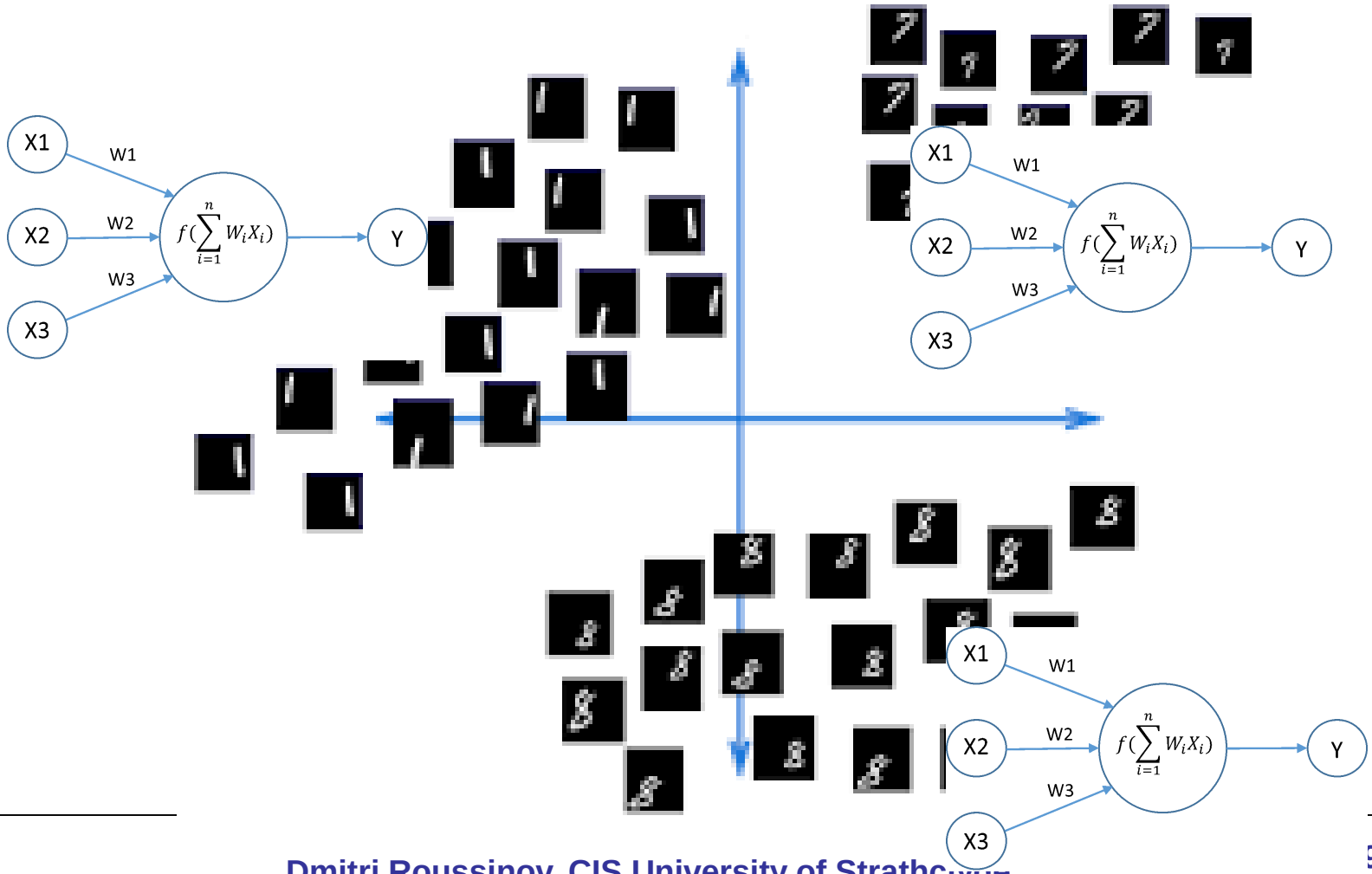


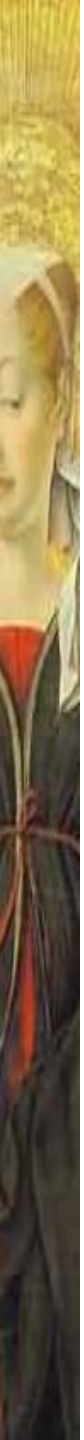
Recall A Single Neuron



- Inputs: *{X1,X2,…}*
- Prediction: *Y = f(a)*
- Activations: *a* = $\sum_{i=1}^{n} W_i X_i$

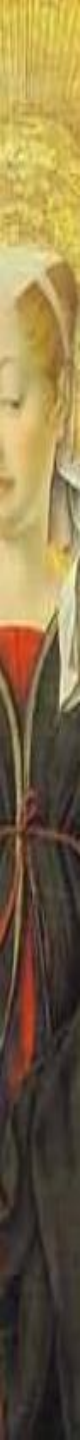
Now Do One Neuron Per Class





Steps Needed

- We use multiple neurons: one per class
- the class with the highest activation wins
- This neural architecture is called “fully connected”
- For this, we need to define our W as a matrix (two dimensional vector)



Pseudocode

Prediction model: // the class with the highest activation wins

    For each class c:

        score[c] = sum (W[c][i]*X[i]) over all features i

    prediction = argmax(score[])

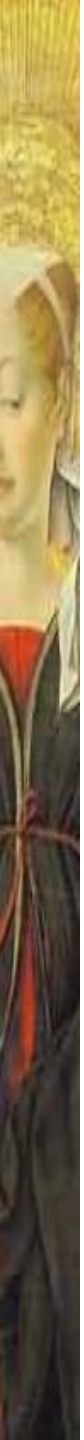
Algorithm:

    For all training examples do:

      For i from 1 to length(W) do:

        W[prediction][i] -= X[i]

        W[label][i] += X[i]



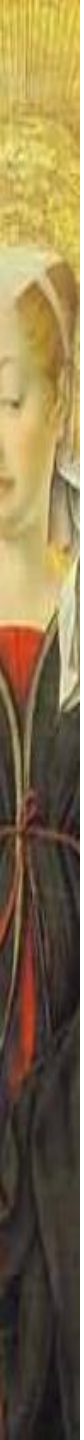
To Note:

- W now is a matrix (two dimensional vector)
  - First index corresponds to the class
  - Second index to the input
- We iterate through all the classes (10 here) and calculate activations
- We return the class that gets the highest activation

  argmax() function used for that

In Java

```
17      static int  image_size = 28 * 28 + 1;
18      static double W [][] = new double [10][image_size];
19
20      public static int output(double X[]) {
21
22          int out = -1;
23          double a_max = -1e10;
24          for(int c = 0; c < 10; c++) {
25
26              check(X.length == W[c].length);
27              double a = 0;
28              for(int i = 0; i < X.length; i++)
29                  a += W[c][i]*X[i];
30              if (a > a_max) {
31                  a_max = a;
32                  out = c;
33              }
34          }
35          check(0 <= out && out < 10);
36          return out;
37      }
```



Exercise: Learn this!

Class	X1	X2	X3	X4
0	1	0	0	1
1	0	1	0	1
2	0	0	1	1