The Multilayer Perceptron

(in other words, a *standard* neural network)

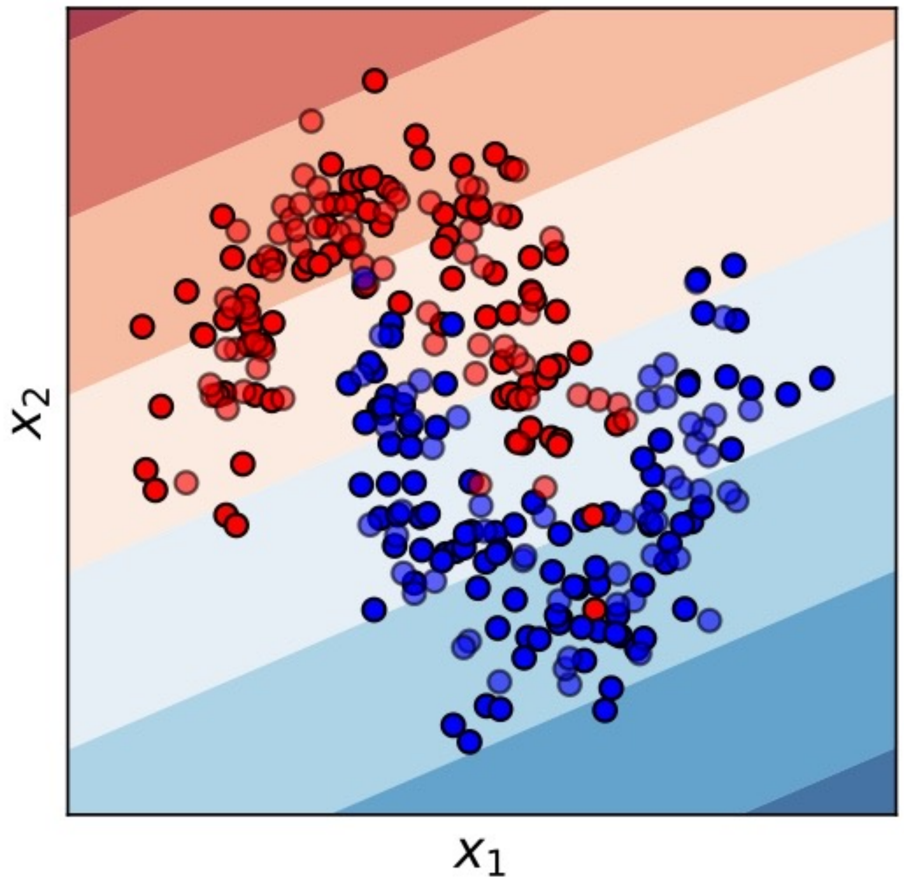
Matthew Engelhard

We need more flexible, non-linear classifiers

- There are many ways to achieve this…

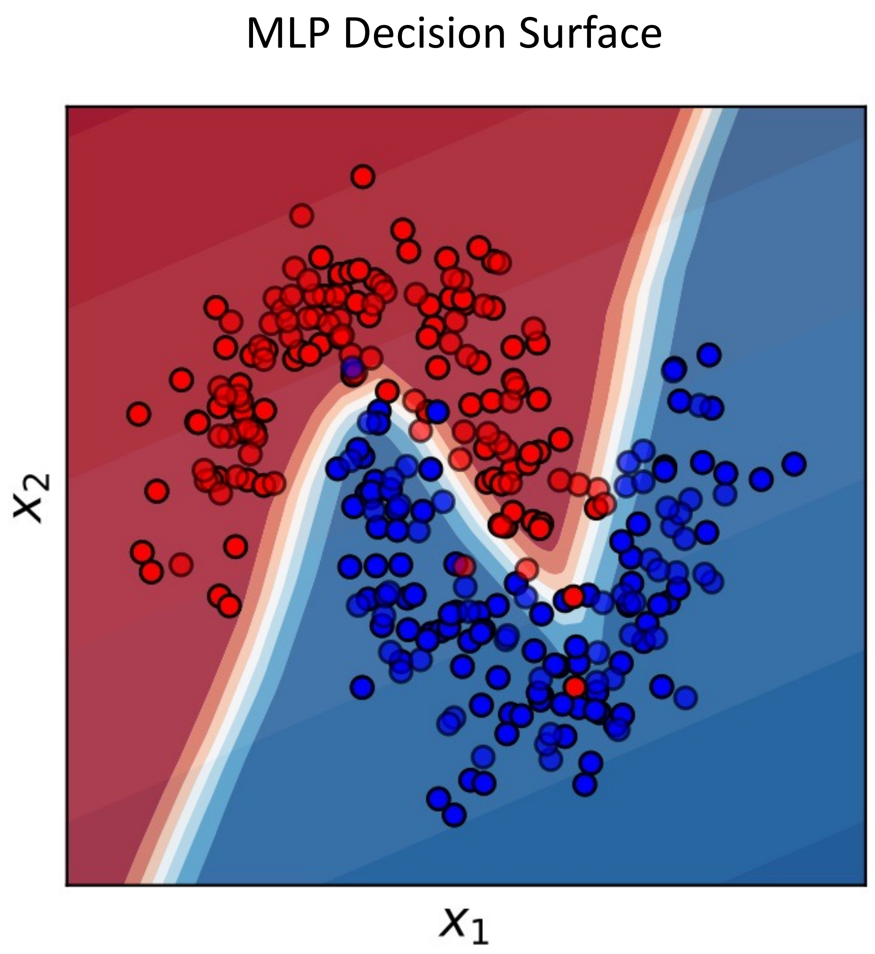
- One of them is to “extend” logistic regression to form a multilayer perceptron (MLP) – in other words, a neural network.

Logistic Regression Decision Surface

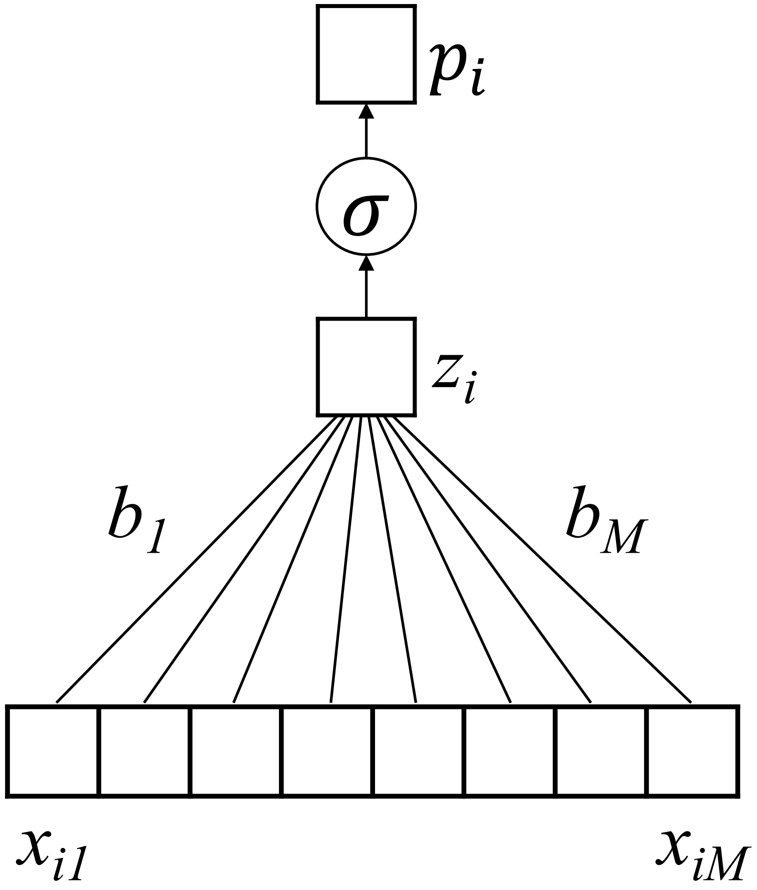


We need more flexible, non-linear classifiers

- There are many ways to achieve this…
- One of them is to “extend” logistic regression to form a multilayer perceptron (MLP) – in other words, a neural network.

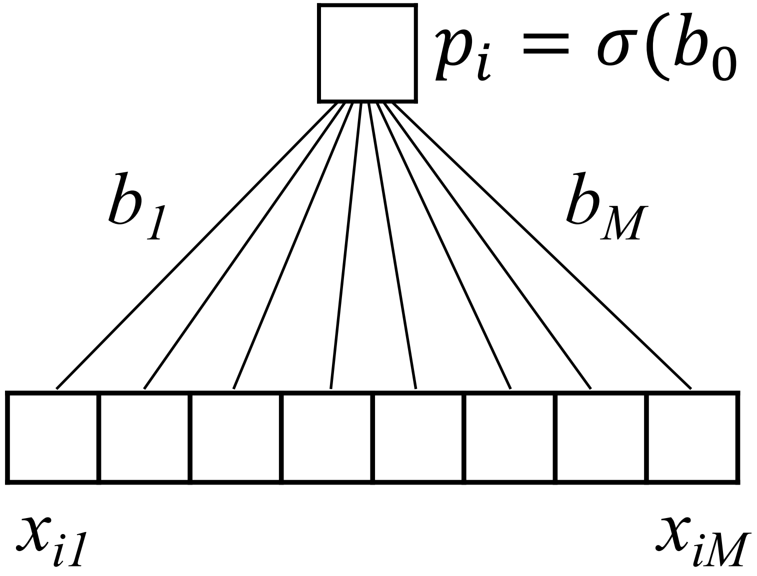


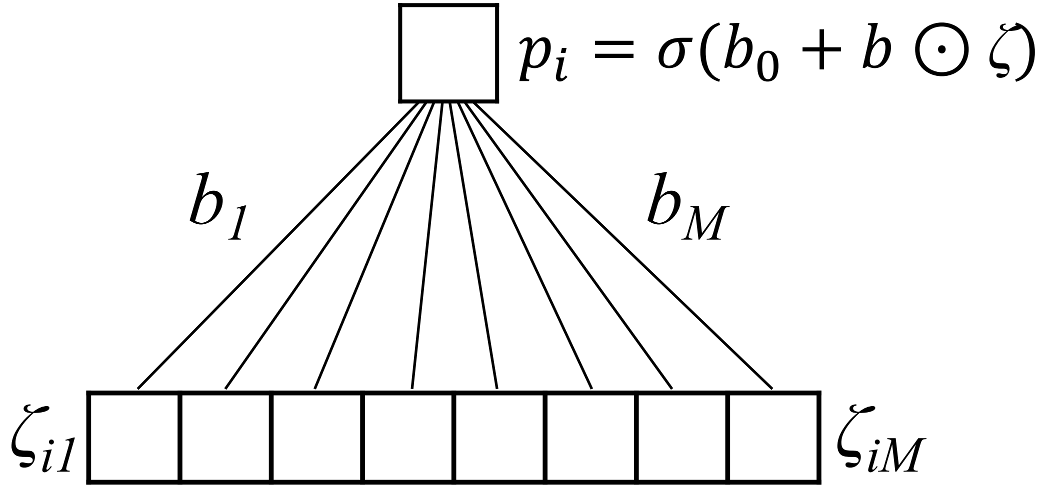
How can we modify logistic regression to learn complex, nonlinear relationships?



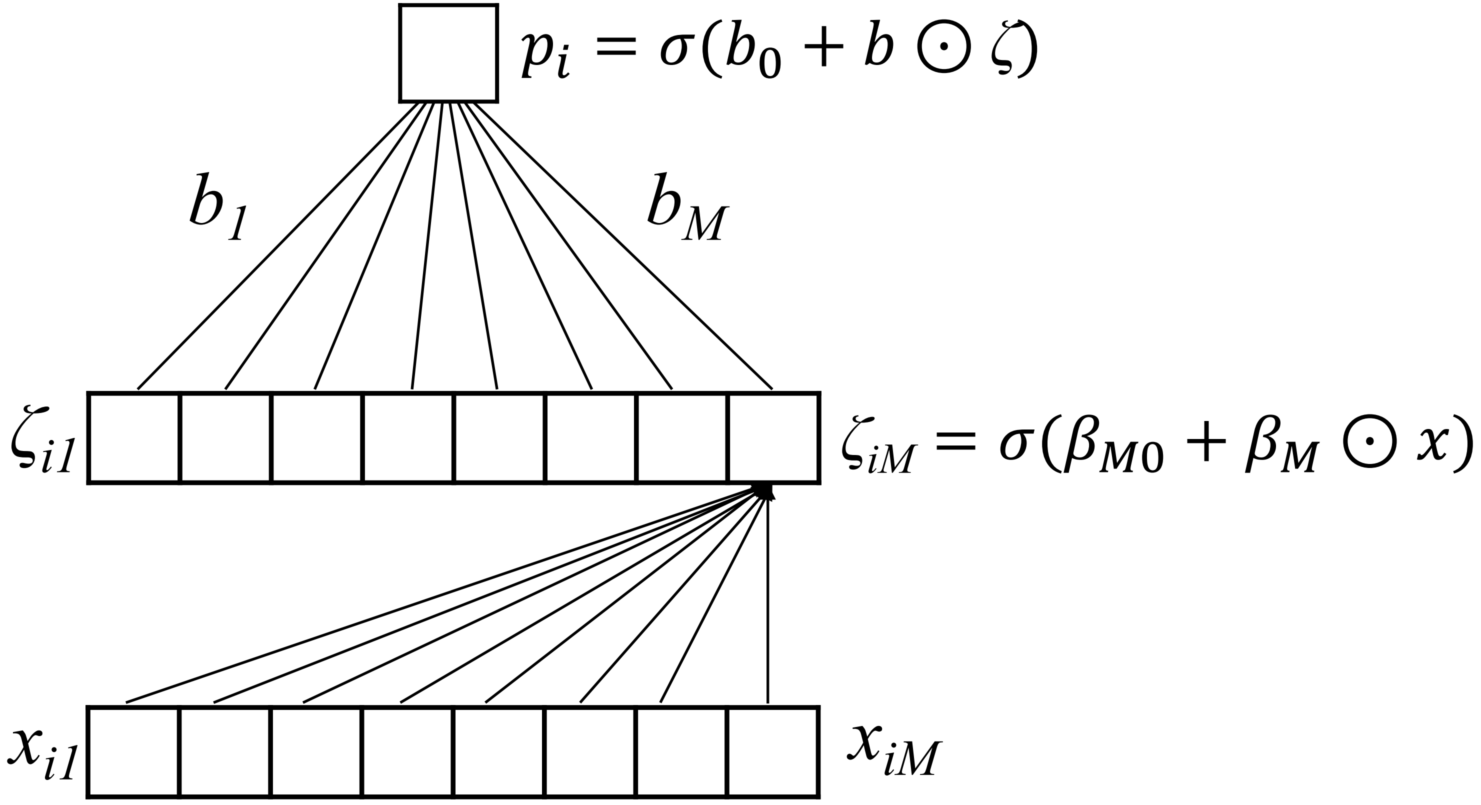
How can we modify logistic regression to learn complex, nonlinear relationships?

$$p_i = \sigma(b_0 + b \odot x)$$



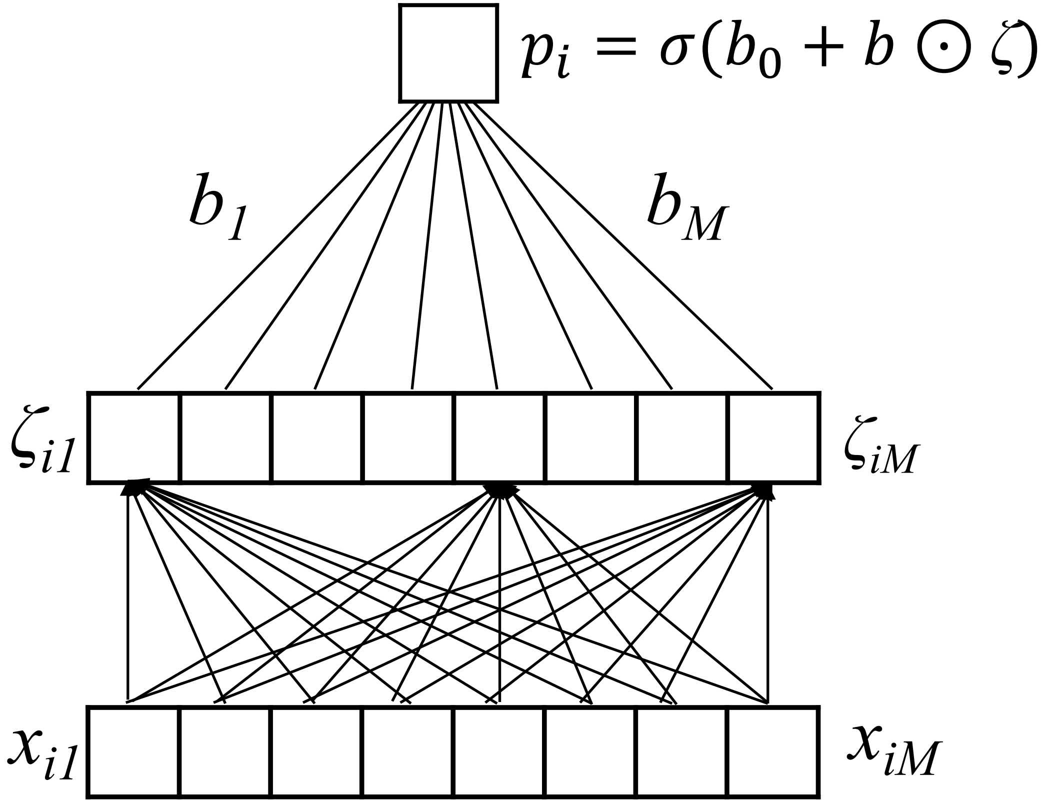


- Instead of predicting p_i directly from our feature vector x, introduce a vector of “latent” features ζ (zeta) that we will use to predict p_i

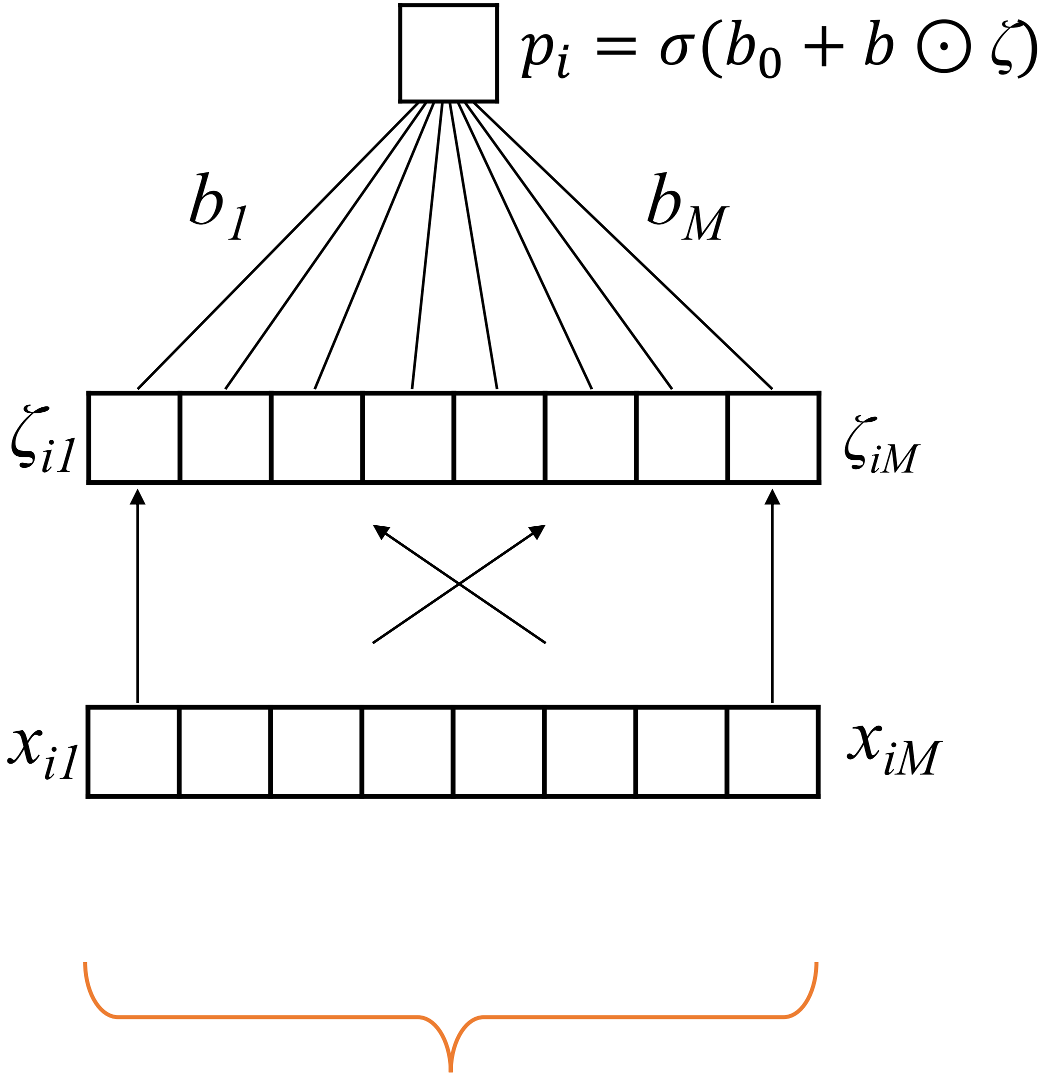


- Instead of predicting p_i directly from our feature vector x, introduce a vector of "latent" features ζ (zeta) that we will use to predict p_i

- Individual elements of ζ will themselves be the output of a logistic-regression-like model based on x



- Instead of predicting p_i directly from our feature vector x, introduce a vector of "latent" features ζ (zeta) that we will use to predict p_i
- Individual elements of ζ will themselves be the output of a logistic-regression-like model based on x
- Since this is true for all elements of ζ, *x* and ζ are said to be "fully connected"

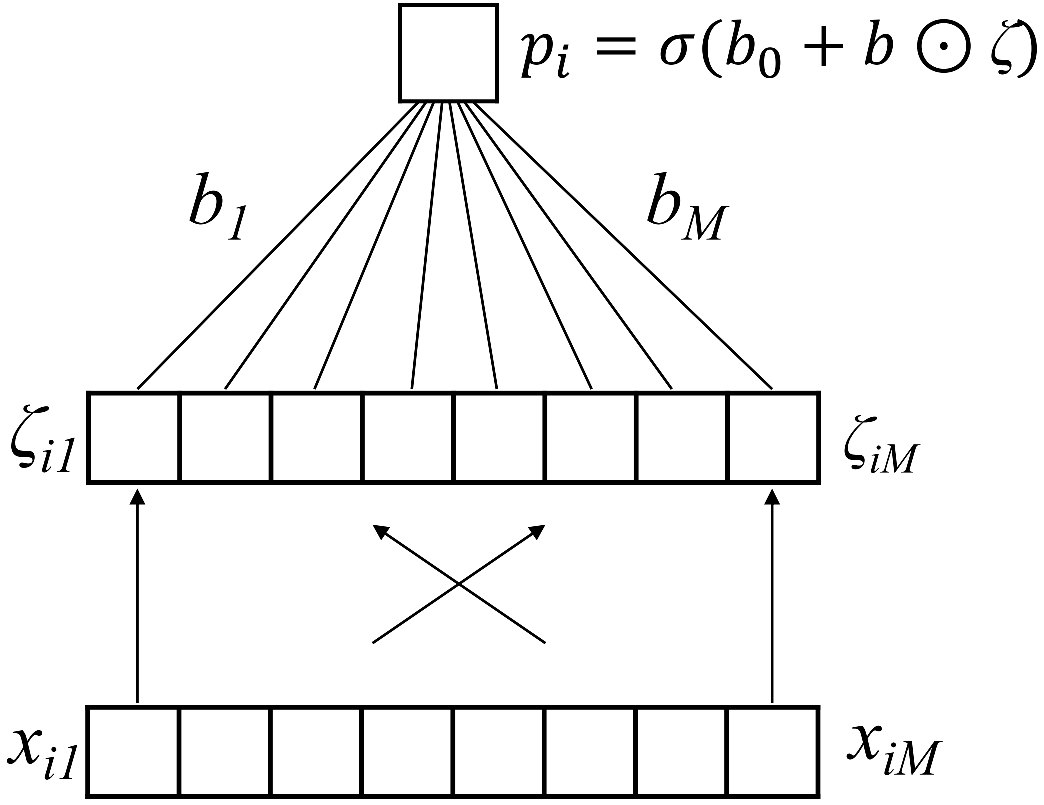


Simplified notation for fully connected layers

- Instead of predicting p_i directly from our feature vector x, introduce a vector of "latent" features ζ (zeta) that we will use to predict p_i

- Individual elements of ζ will themselves be the output of a logistic-regression-like model based on x

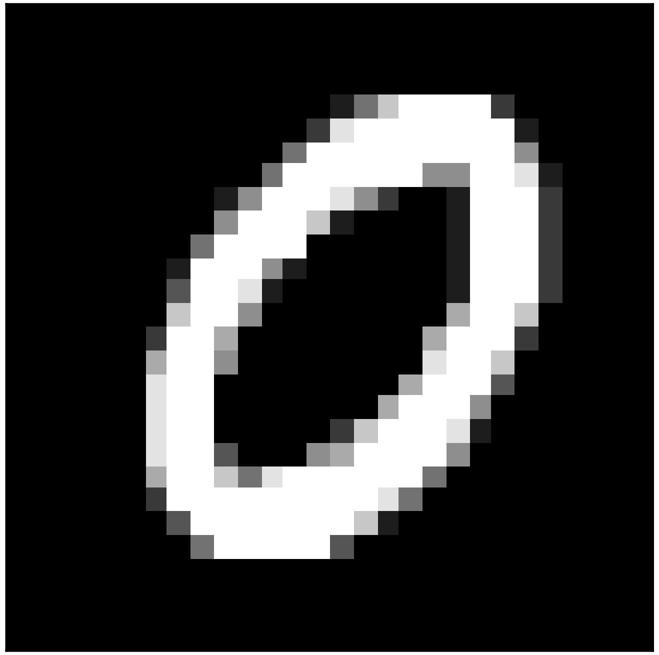
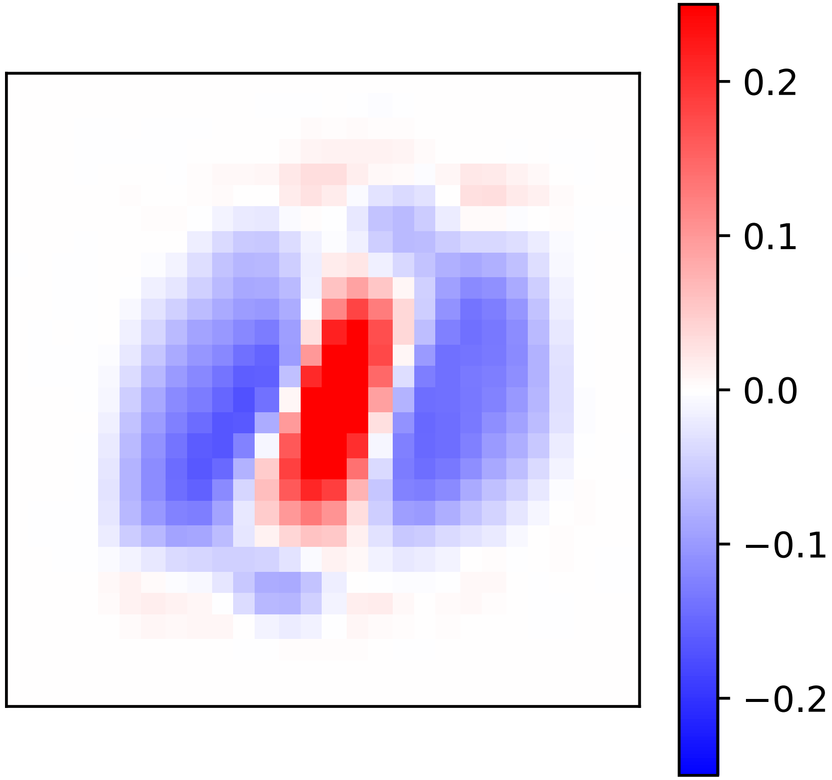
- Since this is true for all elements of ζ, x and ζ are said to be "fully connected"



Since they are neither an input nor an output, the features ζ are said to be a "hidden" layer

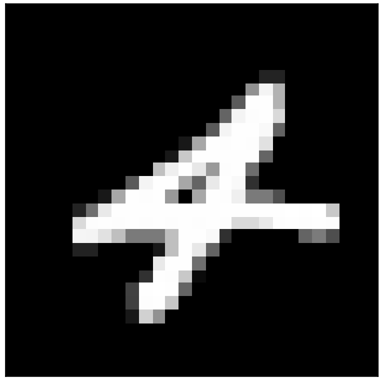
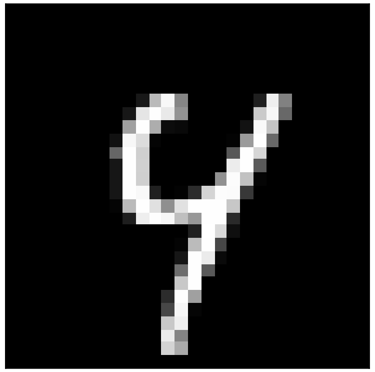
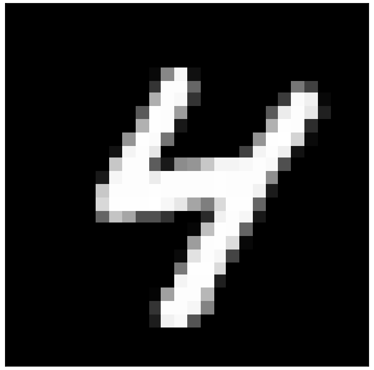
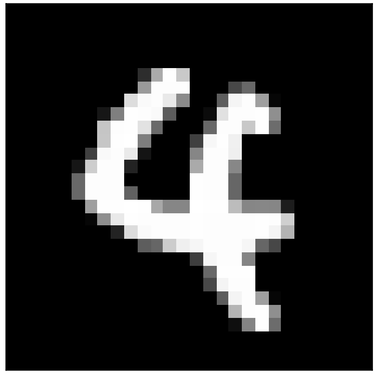
- Instead of predicting p_i directly from our feature vector x, introduce a vector of "latent" features ζ (zeta) that we will use to predict p_i
- Individual elements of ζ will themselves be the output of a logistic-regression-like model based on x
- Since this is true for all elements of ζ, x and ζ are said to be "fully connected"

Why Limit Ourselves to Only One Filter?

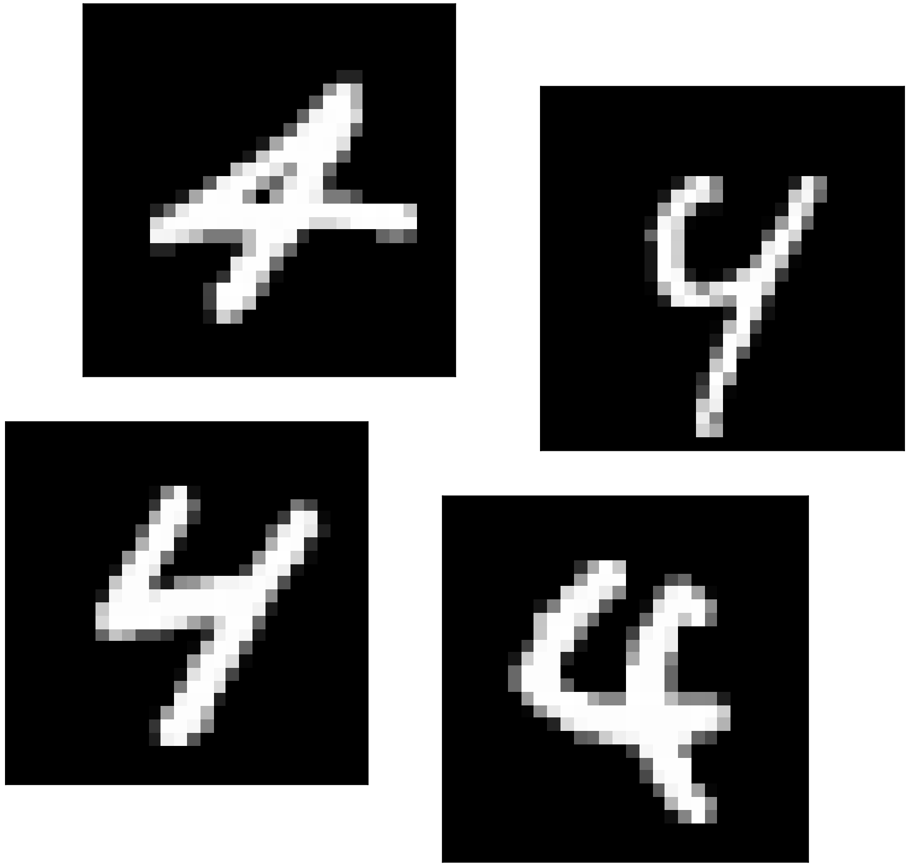
$\sigma($  $\times$ 

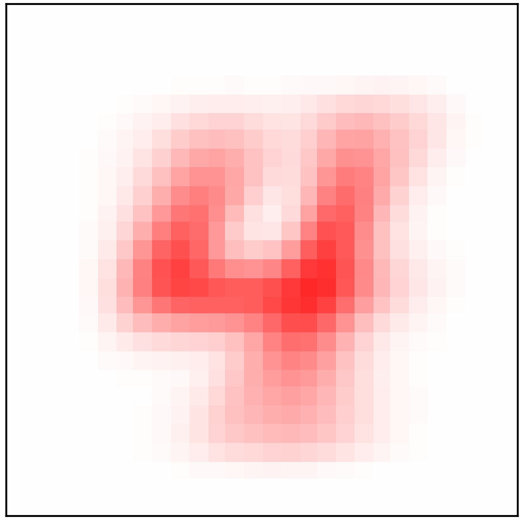
$)$

Return to MNIST:
Many ways of writing "4"

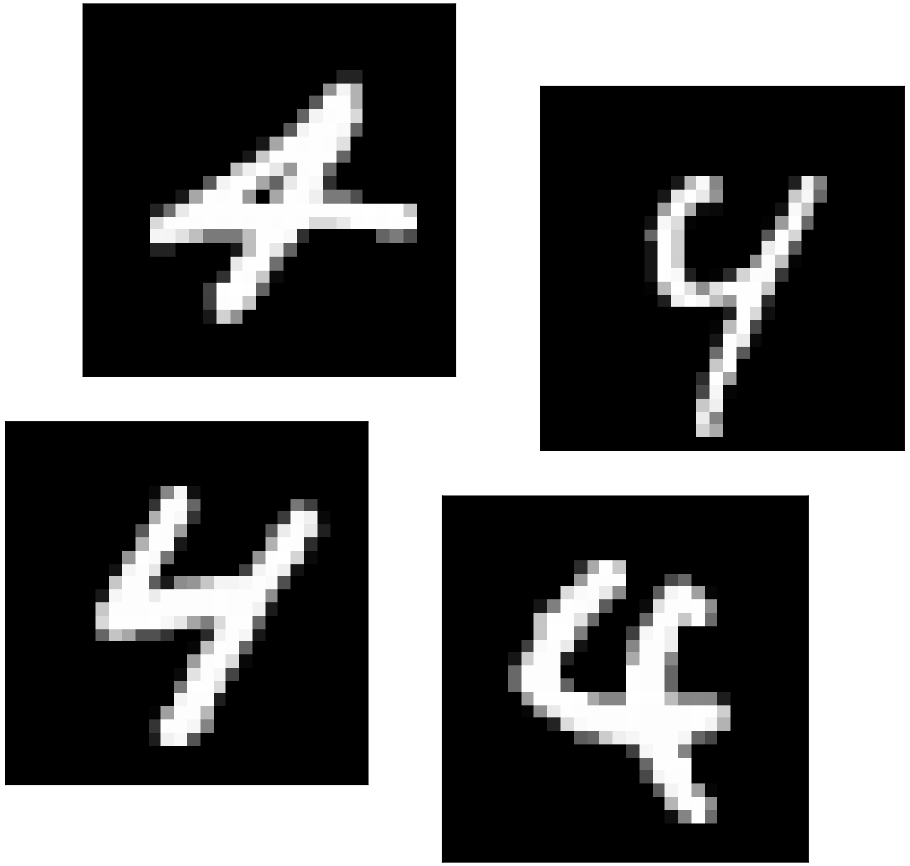
Return to MNIST: Many ways of writing "4"

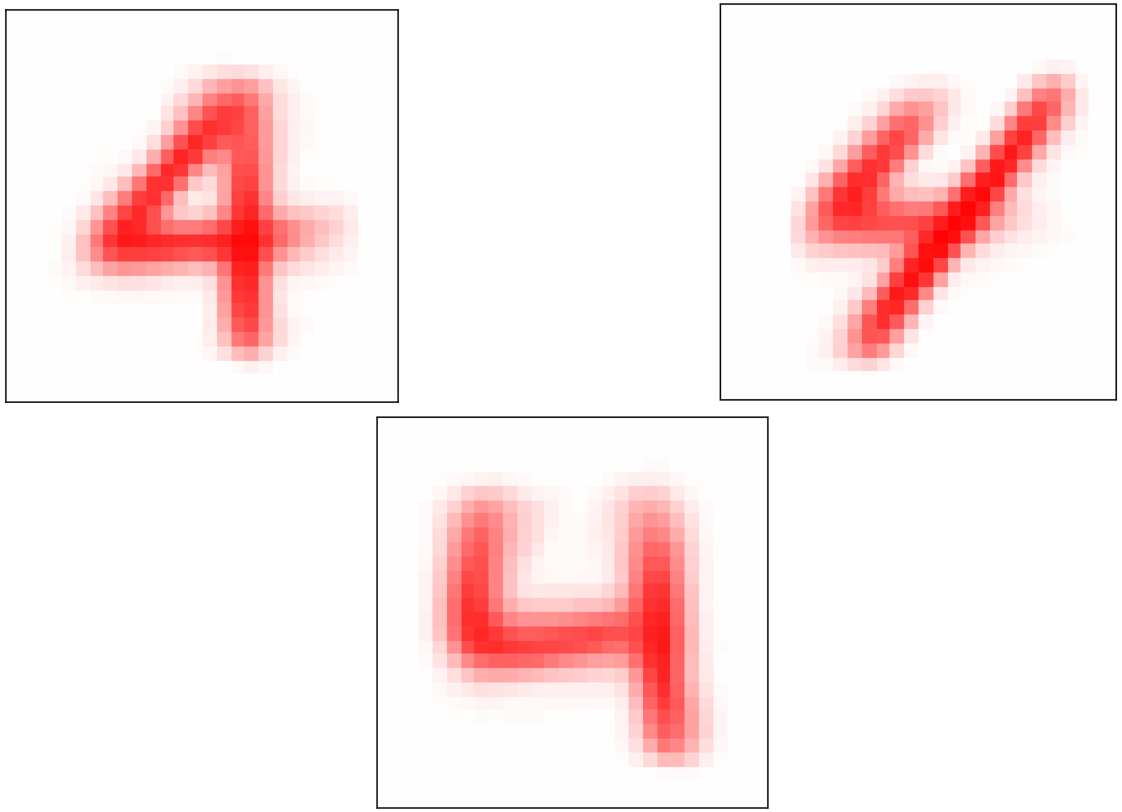




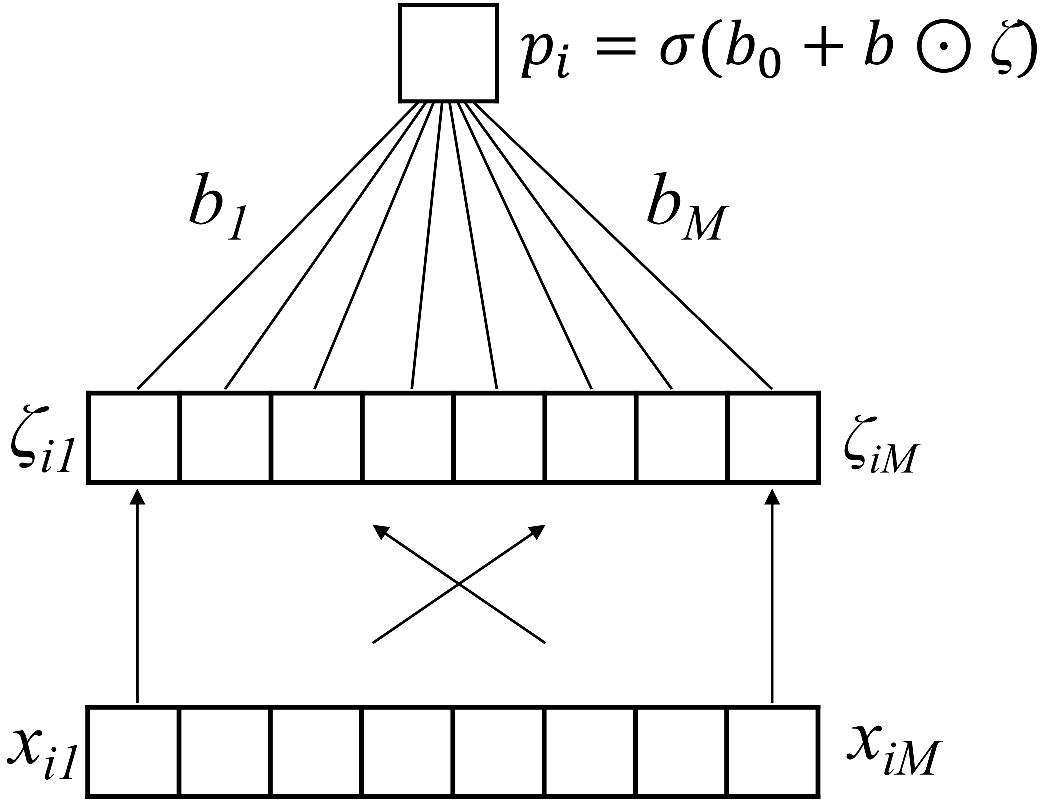
Single Filter (e.g. Logistic Regression/ "Shallow Learning") only uses one filter, looks for the average shape

Return to MNIST: Many ways of writing “4”

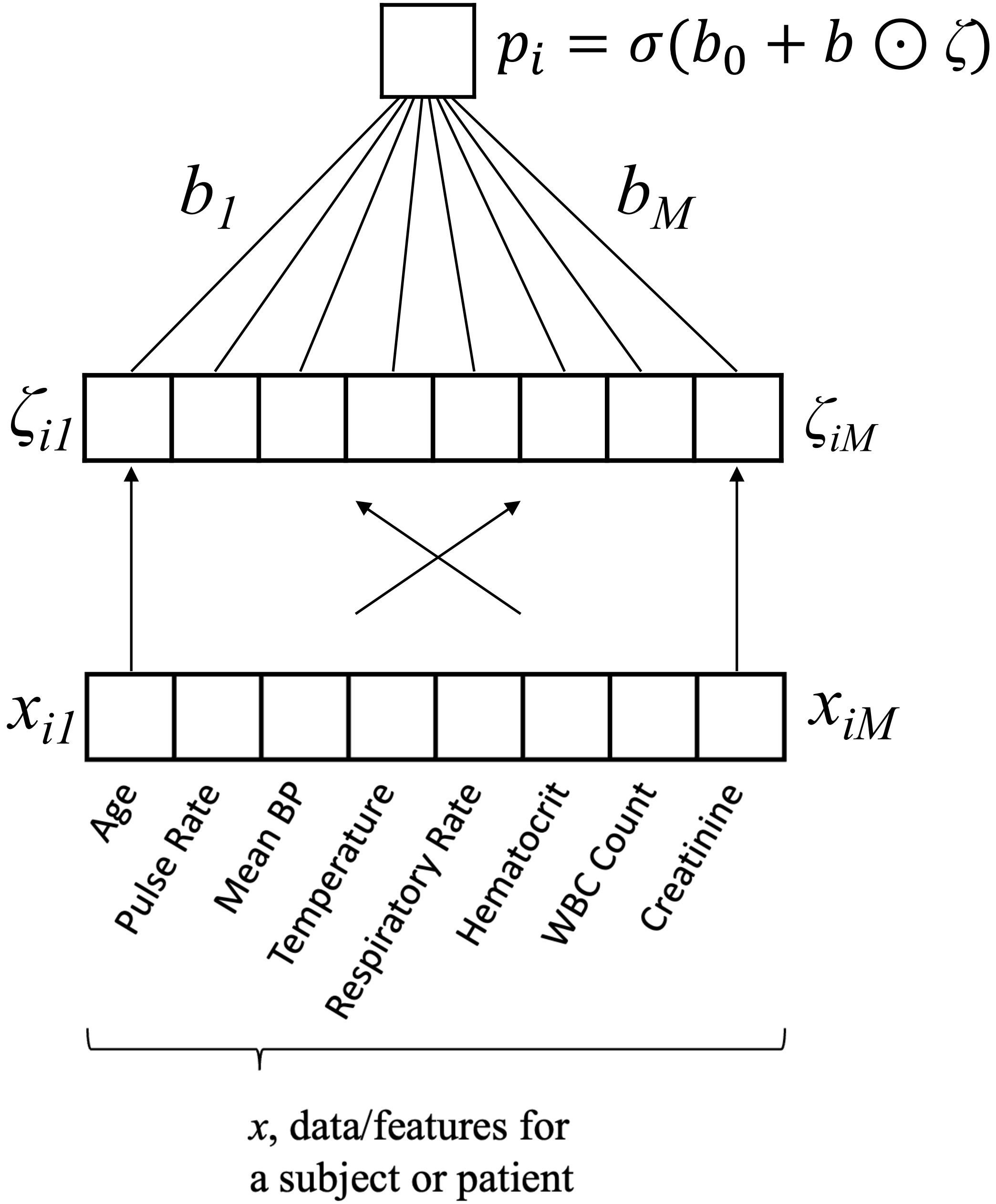




Multiple filters can look for *subtypes* indicative of different ways of writing “4”



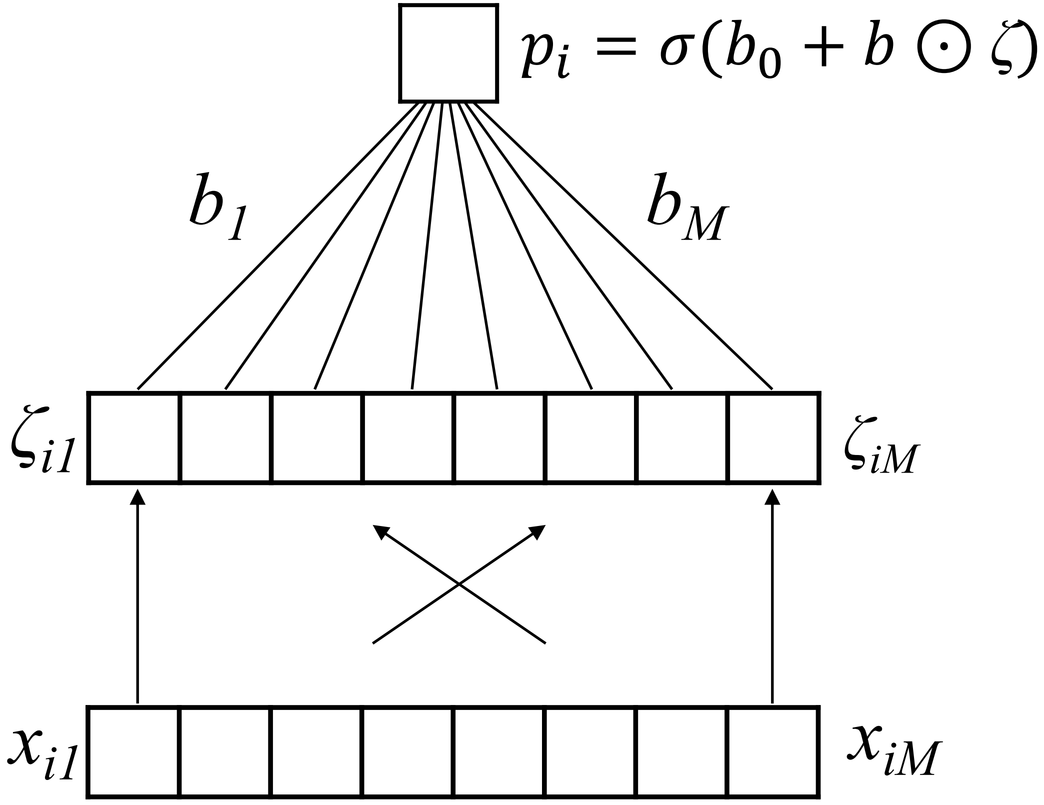
- Each element of ζ_i can be viewed as the output of a single filter applied to x_i
- We then perform logistic regression on the vector of these filter outputs



- Going back to APACHE III, note that both high *and* low values of measurements like age and blood pressure are associated with higher mortality.

- Each element of ζ_i can be viewed as determining how much patient x_i matches a specific risk profile i (sepsis, for example)

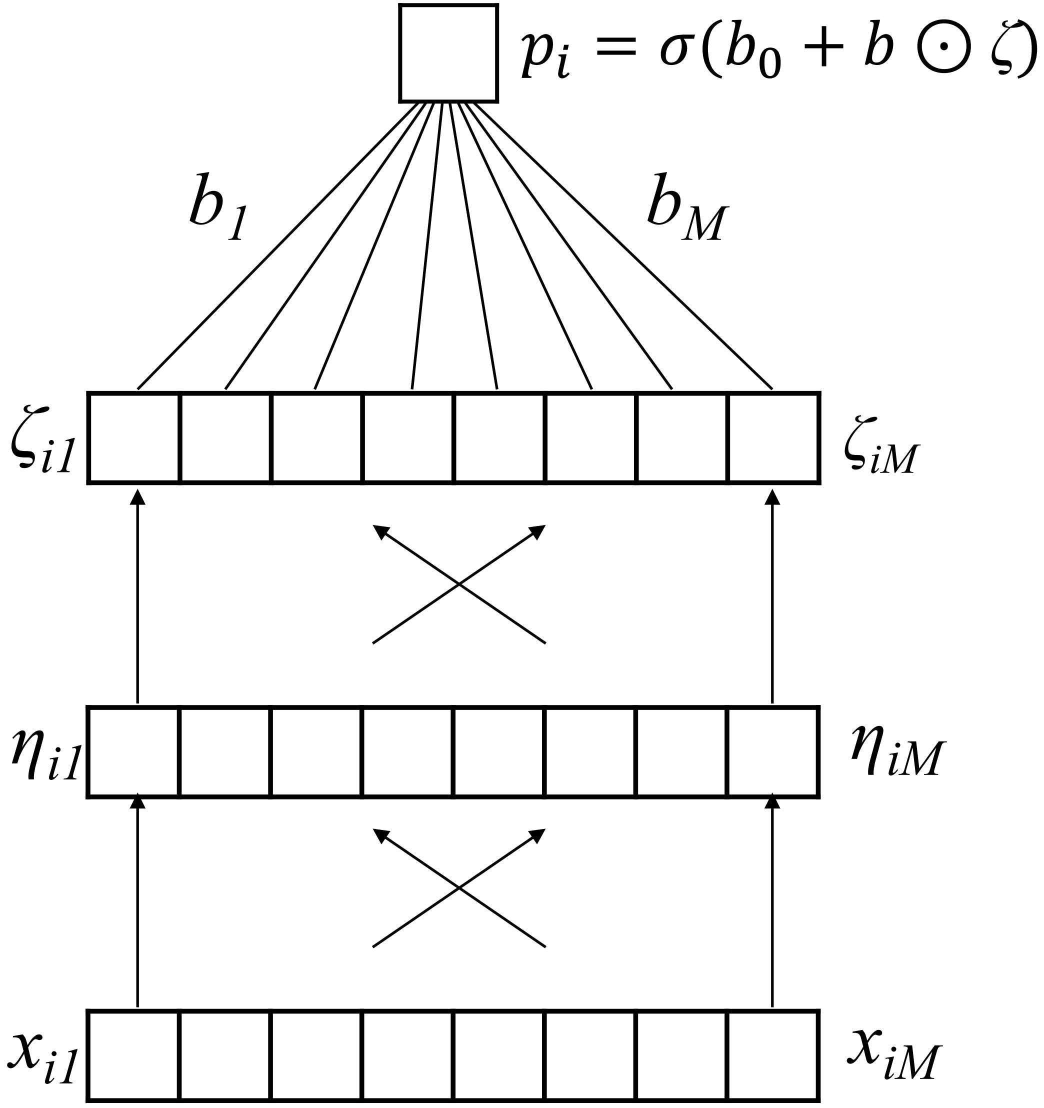
- We then perform logistic regression on the vector of these risk profile matches



Extended logistic regression

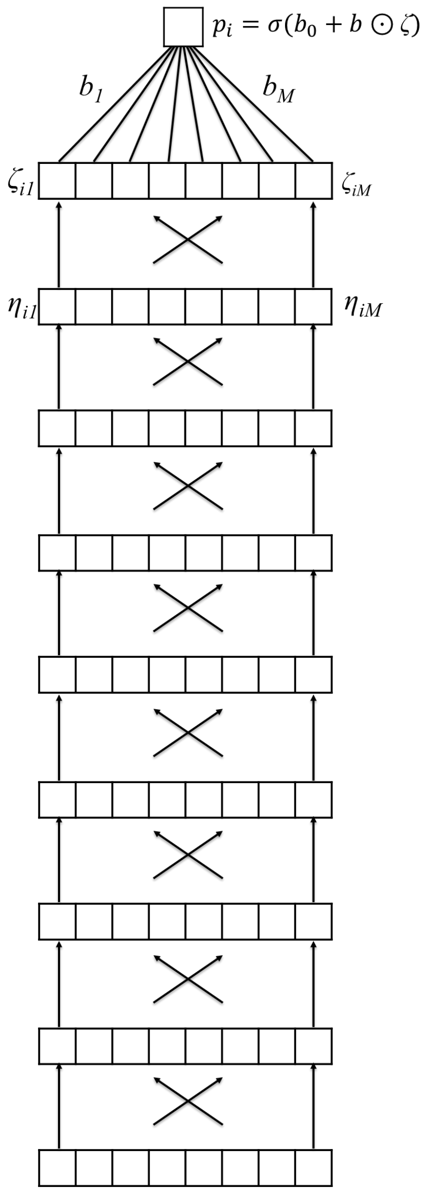
a.k.a.

An MLP with 1 hidden layer ζ



An MLP with 2 hidden layers (η and ζ)

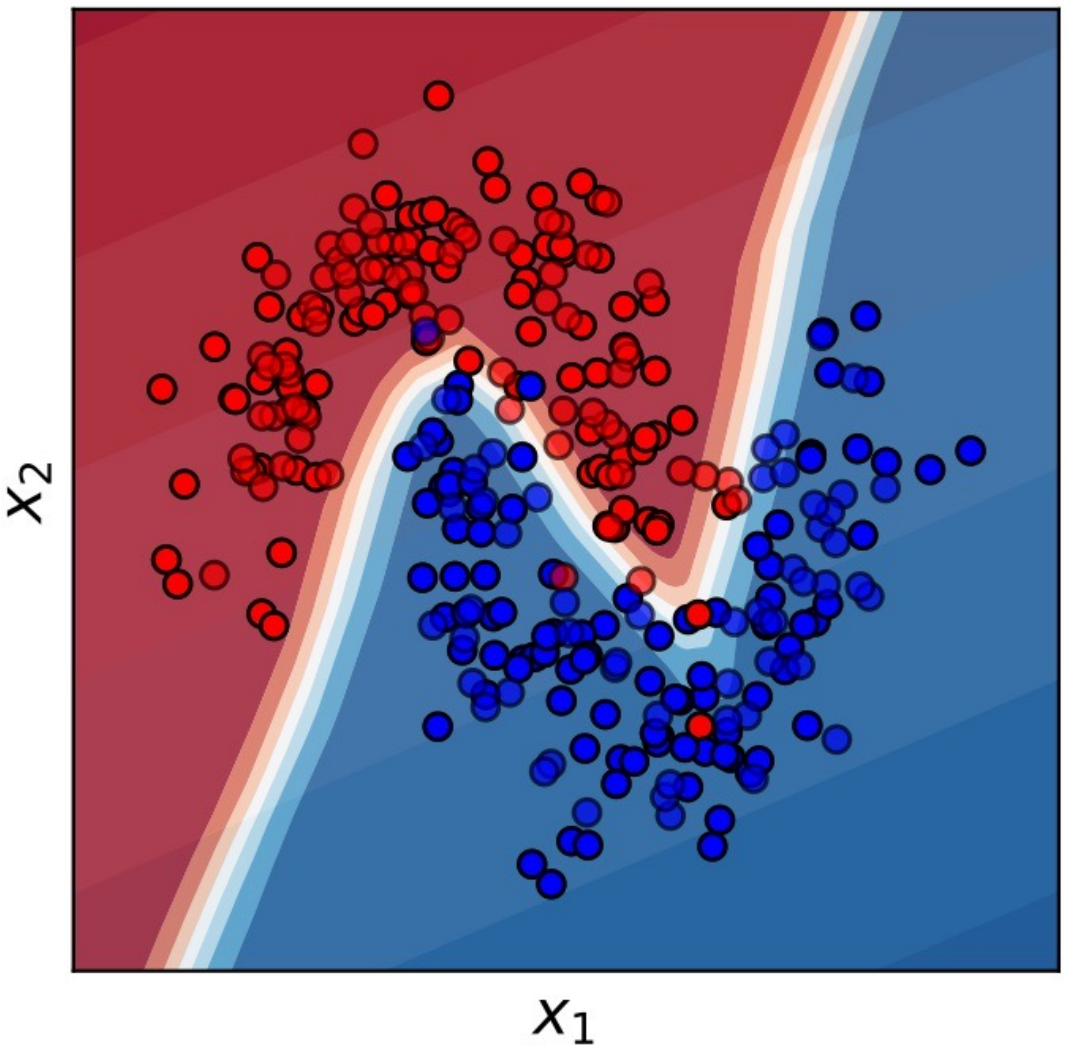
By adding layers, we build a *hierarchy* of increasingly complex features



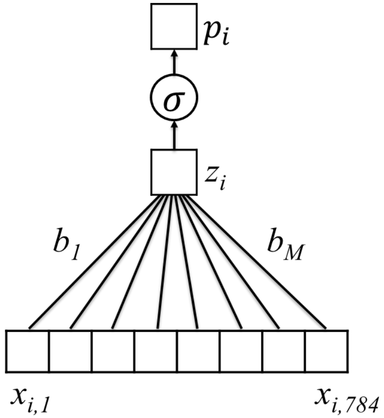
A *deep* MLP with many hidden layers

"deep learning"

Learn Highly Non-Linear Decision Surfaces

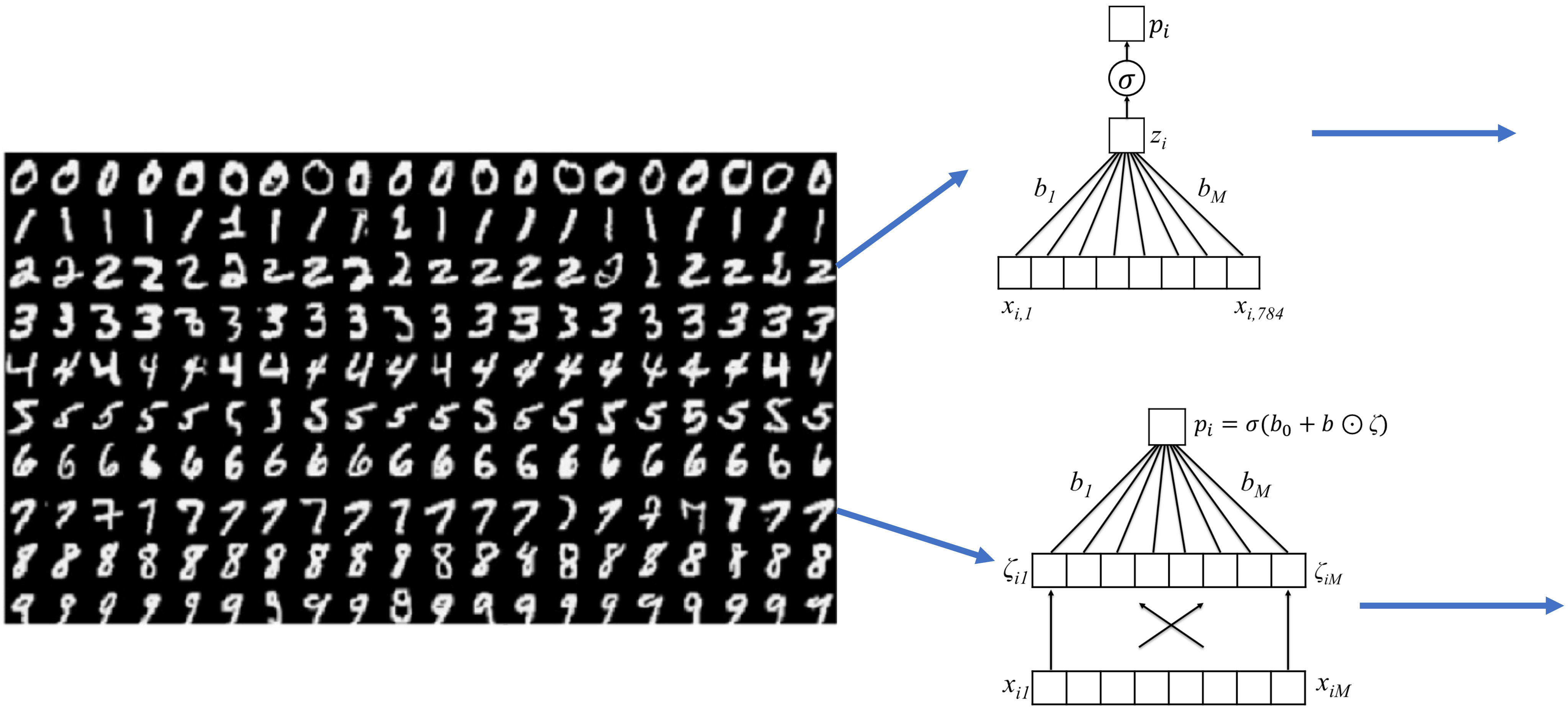


Does this work with MNIST?



~91% Accurate

Does this work with MNIST?



~91% Accurate

~96% Accurate

Summary

- The multilayer perceptron (MLP), also called an artificial neural network (ANN), may be viewed as stacked (layers of) logistic regression models. Logistic regression is applied to latent features, which themselves are the result of earlier logistic regression models.

- We therefore say that the MLP learns a *hierarchy* of features. Each successive level is more complex and/or abstract than the last.

- MLPs can learn highly complex – in fact arbitrarily complex – decision surfaces. However, a large amount of data may be required.