$$
\begin{aligned}
\mathbf{Z}_1 &= \begin{pmatrix}\mathbf{X}\mathbf{W}_0 & \mathbf{1}\end{pmatrix} \\
\mathbf{A}_1 &= f_{\text{ReLU}}(\mathbf{Z}_1) \\
\mathbf{Z}_2 &= \begin{pmatrix}\mathbf{A}_1\mathbf{W}_1 & \mathbf{1}\end{pmatrix} \\
\mathbf{A}_2 &= f_{\text{ReLU}}(\mathbf{Z}_2) \\
&\vdots \\
\hat{\mathbf{Y}} &= \mathbf{A}_{L-1}\mathbf{W}_{L-1}
\end{aligned}
$$

---

89 Definition (Fully-Connected (Dense) Classification Network)
A **fully-connected (dense) classification network** has a softmax output layer.

$$\hat{\mathbf{Y}} \to \boxed{f_{\text{smax}}} \to \hat{\mathbf{P}}$$

$$\hat{\mathbf{P}} = f_{\text{smax}}(\hat{\mathbf{Y}})$$

In the stochastic gradient descent algorithm, a single data point is used to compute a gradient vector. While this strategy for using a single data point has statistical advantages, it is not the most efficient strategy computationally for most compute architectures. Instead, it can be much more efficient to compute the gradient vector using **minibatches** of data points of a given **batch size**.

90 Example (Batch Size)
Determine the most efficient batch size to train logistic regression for the data set `MNIST_train_1000.npz`.

```
from time import time

BS = 10
time_start = time()
hist = model.fit(X,P,epochs=epochs,verbose=0,batch_size=BS)
time_stop = time()
time_elapsed = time_stop - time_start
```

---

91 Example (Non-Convexity of DNN)
Show that a fully-connected neural network with ReLU layers is non-convex. Use the data set `Concrete_train.csv`.
