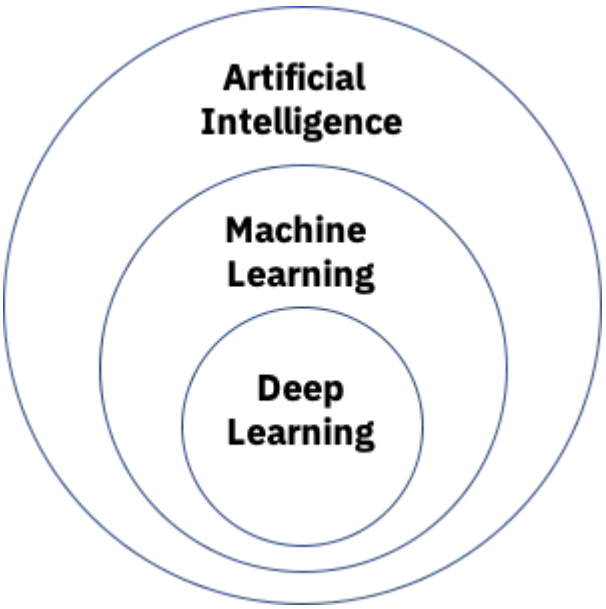
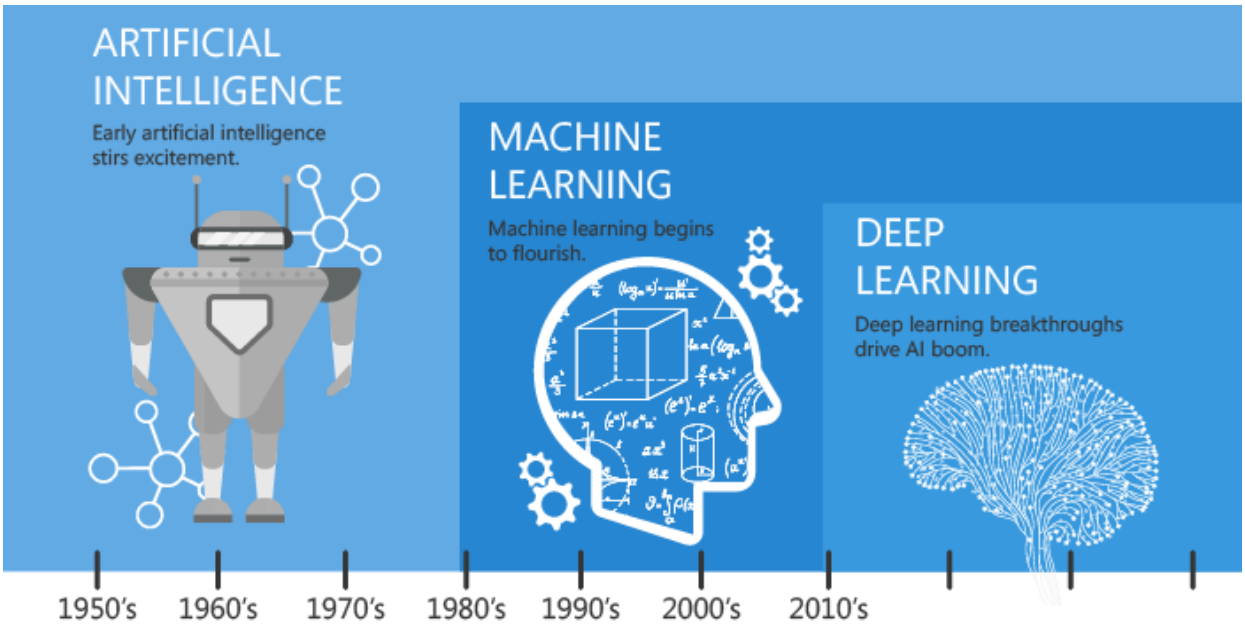
# Lecture 2 - Types of learning is Machine Learning

From last class...




**Artificial Intelligence** A program that can sense, reason, act, and adapt. The two types of AI are: * **Narrow AI** is AI trained and focused to perform specific tasks. * **Strong AI** is made up of Artificial General Intelligence (AGI) and Artificial Super Intelligence (ASI). - Artificial general intelligence (AGI), or general AI, is a theoretical form of AI where a machine would have an intelligence equaled to humans; it would have a self-aware consciousness that has the ability to solve problems, learn, and plan for the future. - Artificial Super Intelligence (ASI) - also known as [superintelligence](https://en.wikipedia.org/wiki/Superintelligence:_Paths,_Dangers,_Strategies) - would surpass the intelligence and ability of the human brain.



**Machine Learning** The machine general ability to solve intelligent tasks by learning from experience/data without being explicitly programmed. Machine learning works with structured data (also referred to as *features*), typically requiring data pre-processing.

**Deep Learning** Subset of machine learning where artificial neural networks learn from large amounts of data. Deep learning eliminates some of data pre-processing that is typically involved with machine learning. These algorithms can ingest and process unstructured data, like text and images, and it automates feature extraction, removing some of the dependency on human experts.

## Representation Learning (or Feature Engineering)

Until the last decade (2010s), the broader class of systems that fell under the label machine learning relied heavily on **feature engineering**. Features are transformations on input data that facilitate a downstream algorithm, like a classifier, to produce correct outcomes on new data.

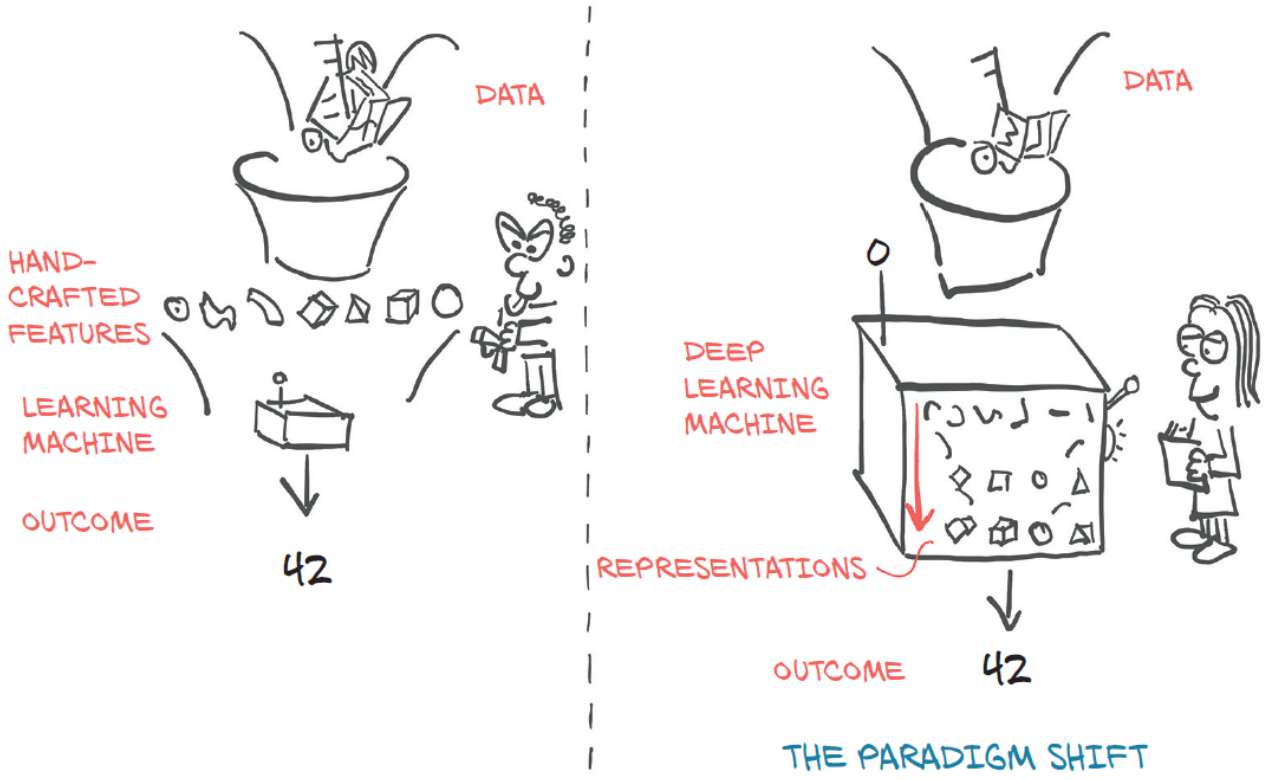
Feature engineering consists of coming up with the right transformations so that the downstream algorithm can solve a task.

For instance, in order to tell ones from zeros in images of handwritten digits, we would come up with a set of filters to estimate the direction of edges over the image, and then train a classifier to predict the correct digit given a distribution of edge directions. Another useful feature could be the number of enclosed holes, as seen in a zero, an eight, and, particularly, loopy twos.

Deep learning, on the other hand, deals with finding such **representations** automatically, from raw data, in order to successfully perform a task. In the ones versus zeros example, filters would be refined during training by iteratively looking at pairs of examples and target labels. This is not to say that feature engineering has no place with deep learning; we often need to inject some form of prior knowledge in a learning system. However, the ability of a neural network to ingest data and extract useful representations on the basis of examples is what makes deep learning so powerful. The focus of deep learning practitioners is not so much on handcrafting those representations, but on operating on a mathematical entity so that it discovers representations from the training data autonomously. Often, these automatically created features are better than those that are handcrafted! As with many disruptive technologies, this fact has led to a change in perspective.

In [8]:
```
from IPython.display import Image
Image('figures/ML vs DL.png', width=900)
#Source: "Deep Learning with PyTorch" by Eli Stevens et al., Manning Publications, 2020
```

Out[8]:




On the left side, we see a practitioner busy defining engineering features and feeding them to a *learning algorithm*; the results on the task will be as good as the features the practitioner engineers.

On the right, with deep learning, the raw data is fed to an algorithm that extracts hierarchical features automatically, guided by the optimization of its own performance on the task; the results will be as good as the ability of the practitioner to drive the algorithm toward its goal.

# Types of Learning

Machine learning algorithms fall into the primary categories:

1. **Supervised Learning** - the model will learn from labeled data
2. **Unsupervised Learning** - the model will learn from unlabeled data
3. **Semi-Supervised Learning** - the model will learn with a training data in which some is labeled, some not, and both are used during training
4. **Reinforcement Learning** - the model will learn which action to take based on reinforcement from environment so to maximize/minimize a reward/penalty on a trial-and-error basis

Other types of learning include:

1. **Multiple Instance Learning** - (type of supervised learning) the model will learn based on multiple-instance labels that have a particular form of imprecision
2. **Active Learning** - (type of supervised learning) the model will learn by obtaining labels online from a user/oracle in an intelligent fashion

3. **Transfer Learning** - the model will learn by transferring learnt knowledge from a similar task.

and many more...

# Supervised Learning

> **Supervised Learning** Learning a mapping from input data to desired output values given labeled training data.

Supervised Learning performs 2 different types of tasks:

1. Classification
2. Regression

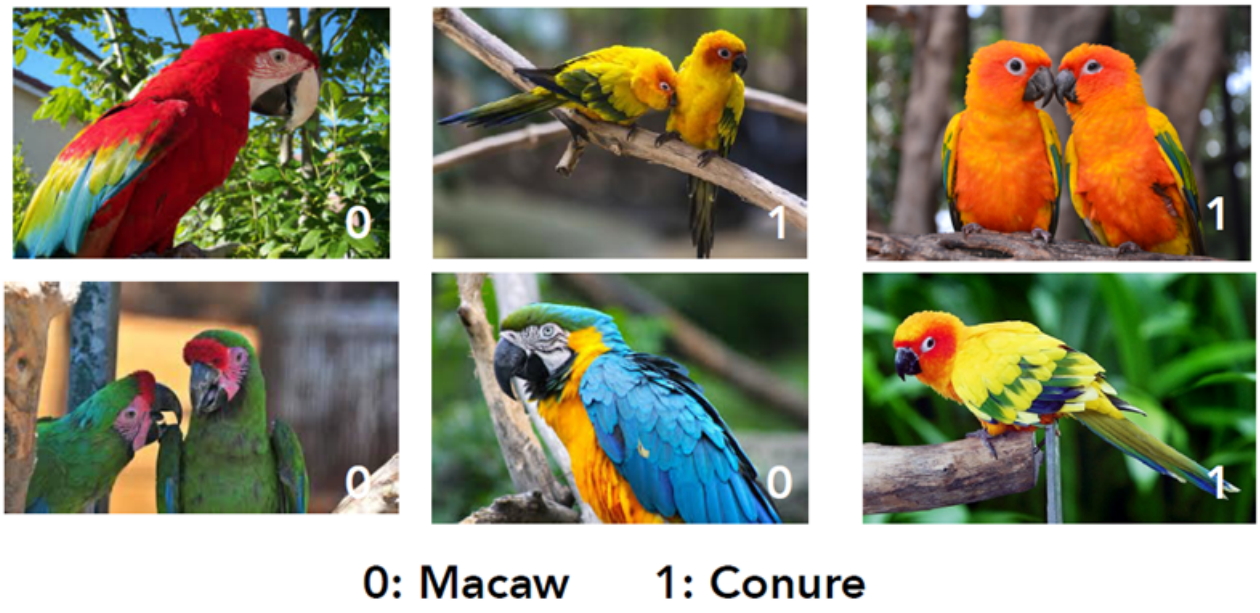
## Classification

**Classification** is a form of predictive modeling approach to characterize the relationship between some collection of observational input data and a set of categorical labels.

Suppose we have training images from two classes, class 0 is macaw and class 1 is conure, and we would like to train a classifier to assign a label to incoming test images whether they belong to class class 0 or class 1.

In [9]:
```
Image('figures/classification.png', width=800)
```

Out[9]:




This is a **classification** example. Each data point was classified into a **discrete class** (either conure or macaw).

Classifiers can further be sub-categorized as **discriminative** or **generative** classifiers.

- A **discriminative** approach for classification is one in which we partition the feature space into regions for each class. Then, when we have a test point, we evaluate in which region it landed

on and classify it accordingly.

- A **generative** approach for classification is one in which we estimate the parameters for distributions that generate the data for each class using Bayesian principles. When we have a test point, we can compute the posterior probability of that point belonging to each class and assign the point to the class with the highest posterior probability.

## Regression

**Regression** is a form of *predictive modeling* approach to characterize the relationship between some collection of observational input data and a continuous desired response.

- A linear regression model is a linear combination of input values.

Consider the example below:

- The goal is to *train* a model that takes in the silhouette images with their correspondent labels (age of the person in the silhouette) and learn a linear relationship between images and age.

In [10]:
```
Image('figures/regression.png', width=800)
```

Out[10]:


- After the model is trained, the **goal** is to be able to *predict* the desired output value of any *new* unlabeled test data.

Applications of regression include: electric/solar power forecast, stock market, inventory investment, etc.

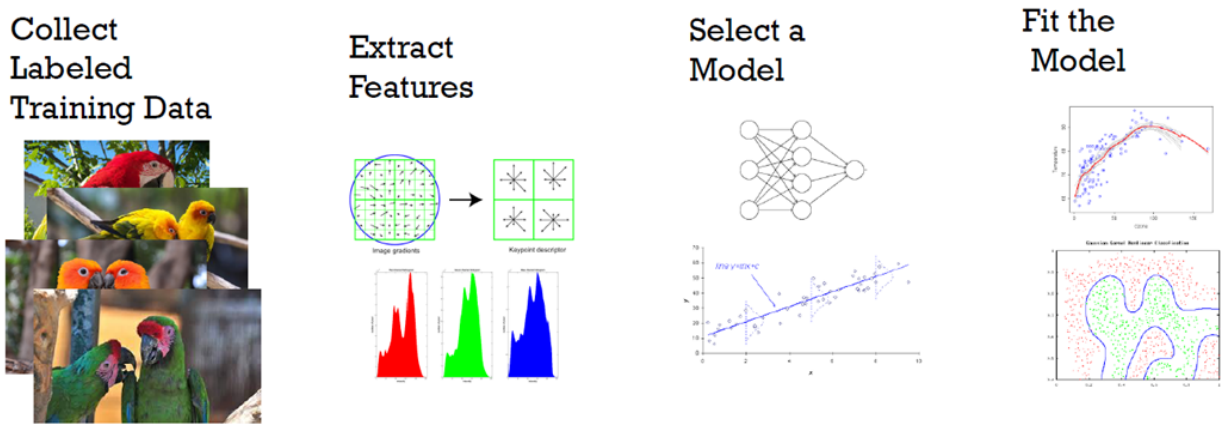
# Typical Flowchart for Supervised Learning

The usual flow (but not always) for supervised learning is:

- **Training stage**
  1. Collect labeled training data - often the most time-consuming and expensive task. This constitutes the **input space**.
  2. Extract features - extract *useful* features from the input (or observational) data such that they have discriminatory information in successfully mapping the desired output. This constitutes the **feature space**.
  3. Select a model - relationship between input data and desired output.
  4. Fit the model - change model parameters (**Learning Algorithm**) in order to meet some **Objective Function**.

In [8]:
```
Image('figures/training.png', width=800)
```

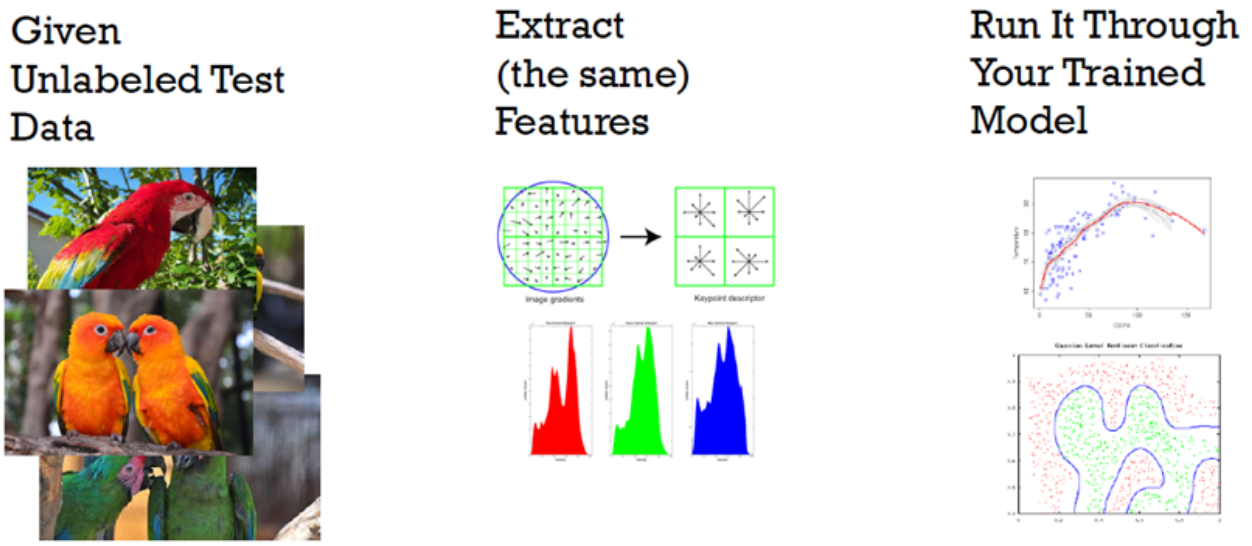
Out[8]:




- **Testing:**
  1. Given unlabeled test data
  2. Extract (the same) features
  3. Run the unlabeled data through the trained model

In [7]:
```
Image('figures/testing.png', width=800)
```

Out[7]:




# Components of a Supervised ML System

Let's open the *virtual whiteboard* to describe the steps of "fitting a model".

From the whiteboard notes, we saw the flowchart for Supervised Learning.

(See whiteboard notes) The system has a **feedback** loop which will make this approach completely **autonomous** without user intervention.

- But yet we have fully control of each component's design choice.

## Challenges

Some of the challenges of supervised learning include:

- How do you know if you have *representative* training data?
- How do you know if you extracted *good* features?
- How do you know if you selected the *right* model?
- How do you know if you selected the *right* objective function?
- How do you know if you trained the model *well*?

Many of these challenges are alleviated (not solved entirely, but helped significantly) with *LOTS AND LOTS* of **data** and a good **experimental design**.

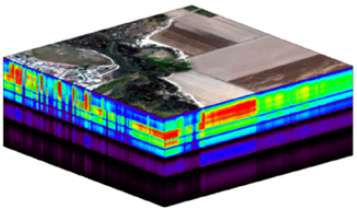
In [11]:
```
Image('figures/PHDcomics.png', width=700)
```

Out[11]:


But, sometimes, obtaining labeled training data is hard, expensive, time consuming and, in some cases, infeasible.

In [12]:
```
Image('figures/NEON.png')
```

Out[12]:



From NEON
neonscience.org

# AI Ethics

IBM's Everyday Ethics proposes five areas of ethical focus are:

1. **Accountability** - AI designers and developers are responsible for considering AI design, development, decision processes, and outcomes. What to expect?
2. **Value Alignment** - AI should be designed to align with the norms and values of your user group in mind.
3. **Explainability** - AI should be designed for humans to easily perceive, detect, and understand its decision process
4. **Fairness** - AI must be designed to minimize bias and promote inclusive representation.
5. **User Data Rights** - AI must be designed to protect user data and preserve the user's power over access and uses.

# Unsupervised Learning

**Unsupervised Learning** Learning structure from data without any labels.

Unsupervised Learning performs 2 different types of tasks:

1. Clustering
2. Dimensionality Reduction & Manifold Learning

## Clustering

**Clustering** algorithms seek to learn, from the properties of the data, an optimal division or discrete labeling of groups (also called *clusters*) of points.

Suppose you collect pictures of the following objects. How many groups would you partition this data into?

In [13]:
```
Image('figures/clustering2.png', width=600)
```

Out[13]:

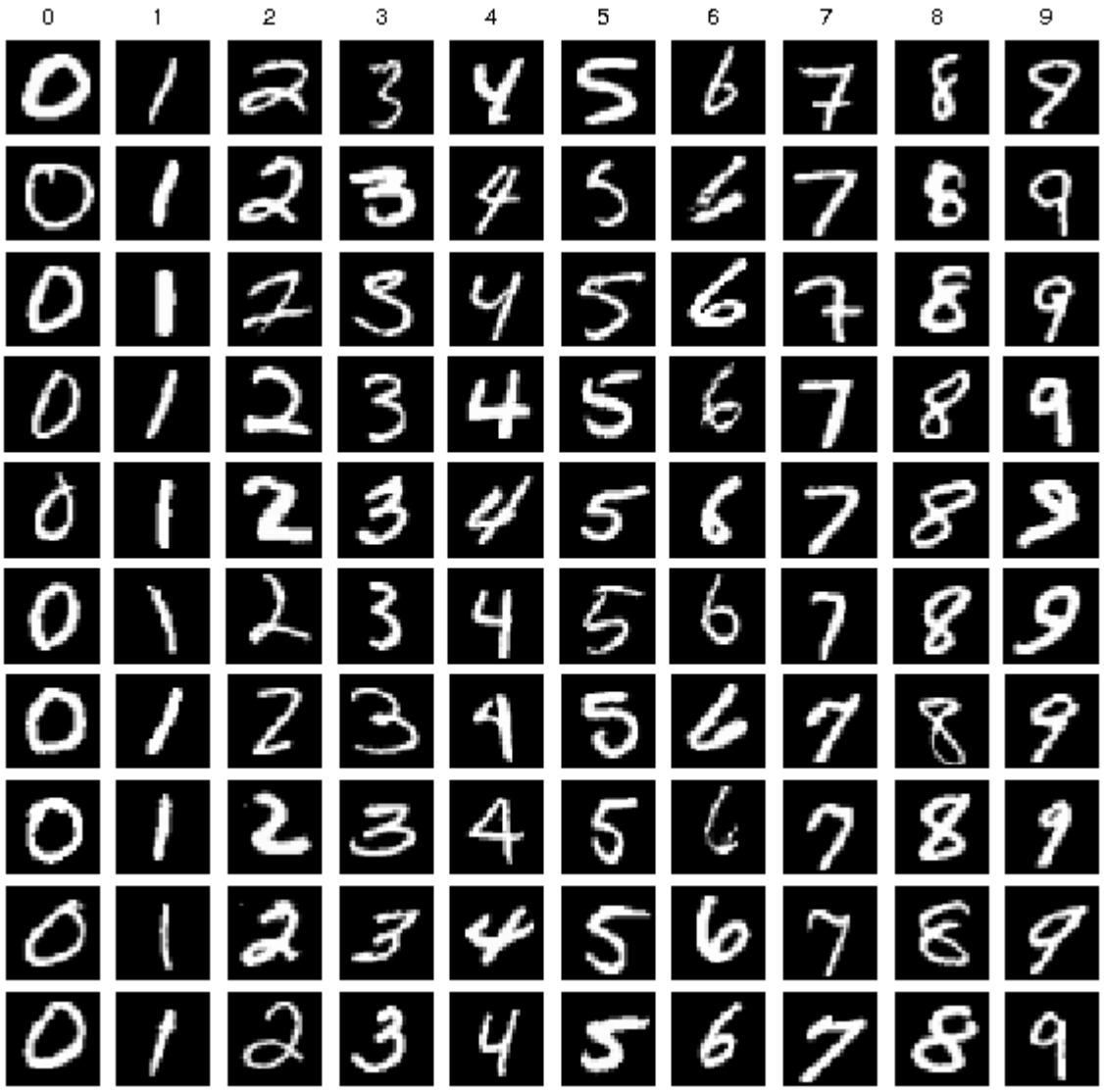

In clustering, we typically *search* for the number of clusters that optimize some relationship between members within a group and members between different groups.

## Manifold Learning

Manifold Learning is *typically* an unsupervised learning approach but can also be considered a supervised learning approach depending on the model choice.

**Manifold learning** algorithms seek to learn an underlying low-dimensional representation of the data.

For example, consider the set of handwritten digits:




Each image is a $28 \times 28$ pixel sized image. This will make a 784-dimensional space. It is impossible to visualize this space. Manifold learning can be used to characterize these images in a much small space, e.g. 2D or 3D.

## Challenges

Some of the challenges of unsupervised learning include:

- How do you *validate* your clustered results?
- How do you know if you selected the *right* similarity measure?
- How do you know if you extracted the *right* dimensions/features?