Lecture 1: Introduction to Machine Learning

Gonzalo De La Torre Parra, Ph.D.

Fall 2021

Overview and Basics

- Overview of standard ML techniques using basic mathematical expressions and MATLAB exercises notebooks.
- Topics: Introduction to concepts of training, testing, and cross-validation. Linear and nonlinear supervised methods in regression and classification including linear discriminant analysis, logistic regression, nearest neighbor, support vector machines, ridge regression, LASSO, elastic net, and neural networks.
- Unsupervised methods including clustering and dimensionality reduction (k-means, GMMs, Hierarchical).
- Additional topics depending on time.

Overview and Basics

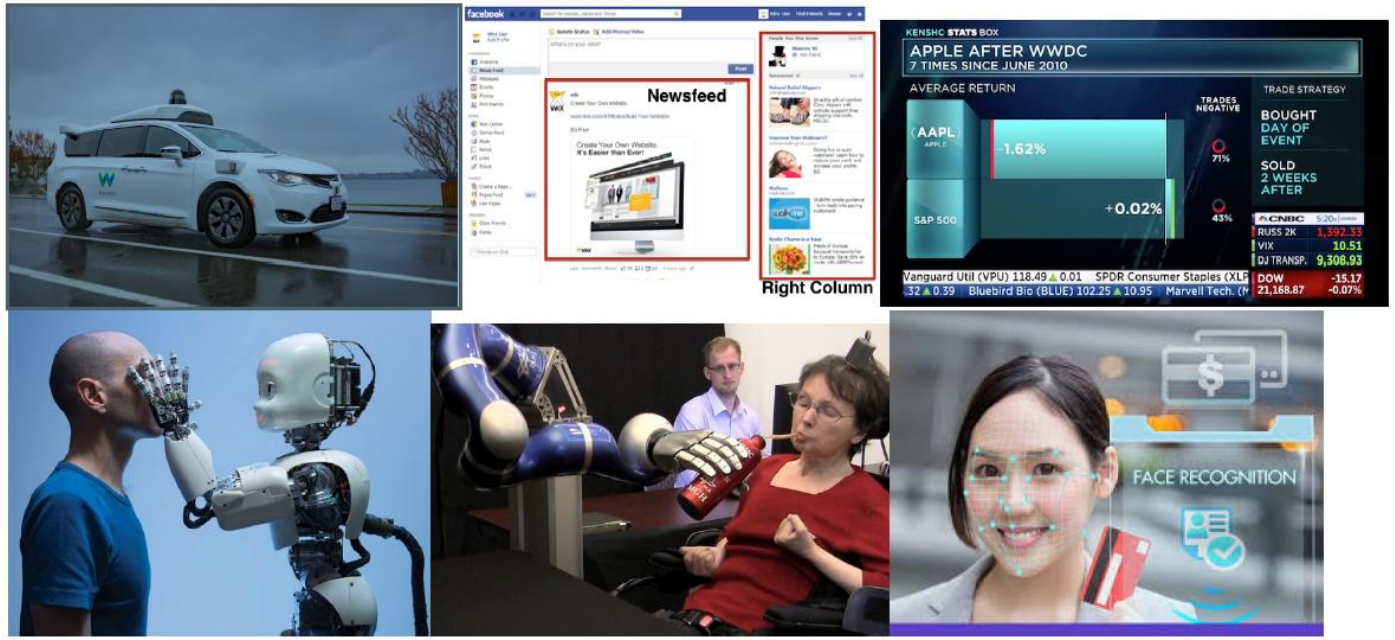
- Mathematics of Machine Learning: Core mathematical ideas used in ML.
- Topics: Probability and statistics, vector spaces, linear algebra, convex optimization, gradient and stochastic gradient descent, ML theory (why ML works).
- Amount of details to be covered depending on interest.

Overview and Basics

- Recent or Advanced Topics: Again, the amount of detail to be covered depends on time.
- Topics: Deep learning, self-supervised learning, reinforcement learning, bandit problems.

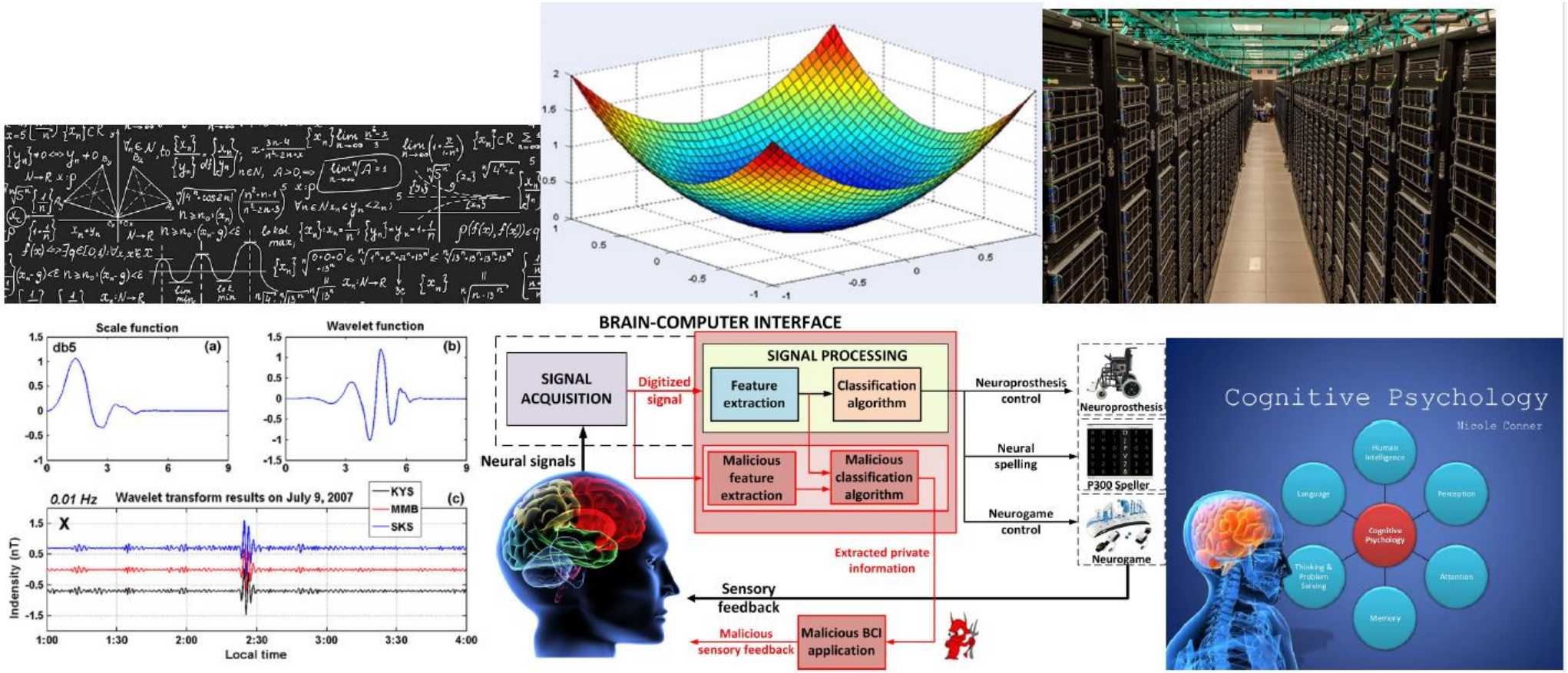
What is Machine Learning?

- Definition: Machine Learning (ML) is a collection of ideas (theory, algorithms, computational techniques) from science, engineering, and social sciences that are or can be used to create artificial intelligence.
- Applications (source: Google images)



Top (L to R): autonomous vehicles, ad optimization, finance.
Bottom (L to R): robotics, brain-computer interfaces, face recognition.

- ML is an interdisciplinary subject: Studied in many departments.



Top (L to R): mathematics and statistics, operation research, computer science.
Bottom (L to R): electrical engineering, neuroscience, humanities.
All departments have a legitimate need to study ML.

ML is an interdisciplinary subject: Studied in many departments.

- Mathematics, Statistics: Provides foundational ideas and techniques.
- Electrical Engineering: Tools from Signal Processing, Control Theory and
- Information Theory are very useful.
- Operations Research: Study optimization problems and their numerical
- solutions.
- Computer Science: What can be computed and how eciently.
- Medicine: Provides important questions and working solutions.
- Humanities: Ideas from Psychology and Sociology helps us to understand how humans think and interact.
- Researchers in one discipline may work on topics from other disciplines.

Data and Two Types of ML Problems

- **The starting point in ML is data.** You are given some data and asked to learn something. The way data is given depends on the type of problem.
- **Supervised ML:** Here you are given n pairs of data points:

  (x1, y1), (x2; y2), … , (xn, yn)

  Problem: x and y have a relationship, you need to discover it.

  Datatype: Both x and y could be vectors or matrices.

  Examples:

  1. Given measurements of some patients (x), predict a quantitative measure of disease (diabetes) progression one year after baseline (y).
  2. Given images of hand-written digits, predict the true digits.
  3. Given measurements of a flower, predict the type of flower.
  4. See the next three slides for samples of such data.

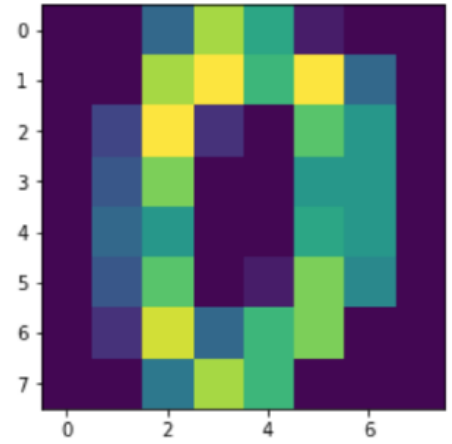
- The diabetes data contains data from 442 patients. It has 10 variables. We need to use those 10 variables (I have shown only 6 of them below) to predict a quantitative measure of disease (diabetes) progression one year after baseline.

Y	AGE	SEX	BMI	BP	S1	S2
151.0	[0.03807591	0.05068012	0.06169621	0.02187235	-0.0442235	-0.03482076]
75.0	[-0.00188202	-0.04464164	-0.05147406	-0.02632783	-0.00844872	-0.01916334]
141.0	[0.08529891	0.05068012	0.04445121	-0.00567061	-0.04559945	-0.03419447]
206.0	[-0.08906294	-0.04464164	-0.01159501	-0.03665645	0.01219057	0.02499059]
135.0	[0.00538306	-0.04464164	-0.03638469	0.02187235	0.00393485	0.01559614]
97.0	[-0.09269548	-0.04464164	-0.04069594	-0.01944209	-0.06899065	-0.07928784]
138.0	[-0.04547248	0.05068012	-0.04716281	-0.01599922	-0.04009564	-0.02480001]
63.0	[0.06350368	0.05068012	-0.00189471	0.06662967	0.09061988	0.10891438]
110.0	[0.04170844	0.05068012	0.06169621	-0.04009932	-0.01395254	0.00620169]
310.0	[-0.07090025	-0.04464164	0.03906215	-0.03321358	-0.01257658	-0.03450761]

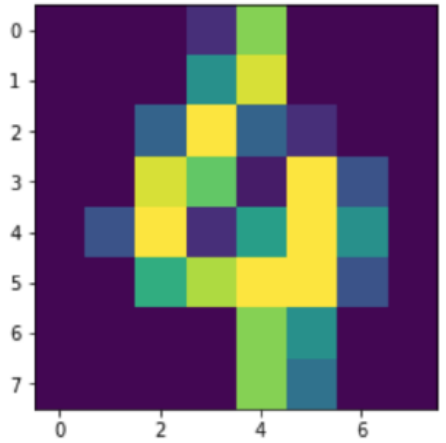
- The data is mean-shifted and normalized. That is why AGE is less than 1 and negative.
- **Since what we want to predict, i.e., Y , is a real number, this supervised ML problem is called a problem in REGRESSION.**

- The digits data is made up of 1797 8 8 images. We need to use the images (stored as matrices) to predict the true digit.

```
[[ 0.  0.  5. 13.  9.  1.  0.  0.]
 [ 0.  0. 13. 15. 10. 15.  5.  0.]
 [ 0.  3. 15.  2.  0. 11.  8.  0.]
 [ 0.  4. 12.  0.  0.  8.  8.  0.]
 [ 0.  5.  8.  0.  0.  9.  8.  0.]
 [ 0.  4. 11.  0.  1. 12.  7.  0.]
 [ 0.  2. 14.  5. 10. 12.  0.  0.]
 [ 0.  0.  6. 13. 10.  0.  0.  0.]]
```




```
[[ 0.  0.  0.  2. 13.  0.  0.  0.]
 [ 0.  0.  0.  8. 15.  0.  0.  0.]
 [ 0.  0.  5. 16.  5.  2.  0.  0.]
 [ 0.  0. 15. 12.  1. 16.  4.  0.]
 [ 0.  4. 16.  2.  9. 16.  8.  0.]
 [ 0.  0. 10. 14. 16. 16.  4.  0.]
 [ 0.  0.  0.  0. 13.  8.  0.  0.]
 [ 0.  0.  0.  0. 13.  6.  0.  0.]]
```



- The Xs are the 8 x 8 matrices and Ys are the actual true digits: Y = 0 and Y = 4.
- Since what we want to predict, i.e., Y , is a discrete-valued quantity or is a label, this supervised ML problem is called CLASSIFICATION.

The **Iris data** sets consists of 150 data points of 3 different types of irises (Setosa, Versicolour, and Virginica). There are four features per sample: Sepal Length, Sepal Width, Petal Length and Petal Width.

Y	Sepal Length	Sepal Width	Petal Length	Petal Width
0	[5.1	3.5	1.4	0.2]
0	[4.9	3.	1.4	0.2]
0	[4.7	3.2	1.3	0.2]
0	[4.6	3.1	1.5	0.2]
1	[6.4	3.2	4.5	1.5]
1	[6.9	3.1	4.9	1.5]
1	[5.5	2.3	4.	1.3]
1	[6.5	2.8	4.6	1.5]
2	[5.8	2.7	5.1	1.9]
2	[7.1	3.	5.9	2.1]
2	[6.3	2.9	5.6	1.8]
2	[6.5	3.	5.8	2.2]

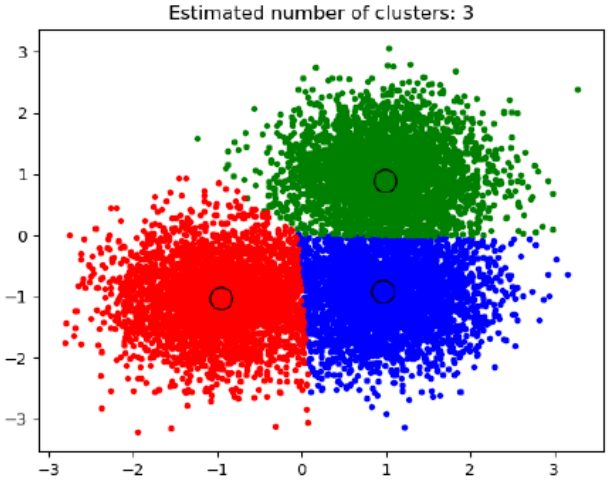
The class labels are encoded as Setosa ($Y = 0$), Versicolour ($Y = 1$), and Virginica ($Y = 2$).

Again, since what we want to predict, i.e., Y, is a discrete-valued quantity or is a label, this supervised ML problem is called CLASSIFICATION.

Unsupervised ML: Here you are given n **data** points:

$$x_1, x_2, \ldots, x_n.$$

Problem: There is some pattern in x, you need to discover it.
Example 1: One important unsupervised problem is that of **clustering**. Given some collection of data, you want to divide the data into groups. The grouping could be based on physical location as shown below.



Example 2 Document classification: Given a set of documents, divide them according to topics or subjects using keywords used in them. Think of finding trends in ML by automatically reading ML research papers.

Training for Supervised ML

After data, the next most important concept in ML is Training. Recall that we are given data

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n).$$

The process of finding a good predictor is called Training.

What does a predictor looks like?:Let $x \in \mathbb{R}^d$ and $y \in \mathbb{R}$. Let h be a function that maps $d-$dimensional vectors to real numbers:

$$h : \mathbb{R}^d \to \mathbb{R}.$$

A predictor is simply a funtion that maps a value in the domain of x to a value in the domain of y; here from $\mathbb{R}^d \to \mathbb{R}$.

Performance of a predictor at a data point: If we choose a given function h to be our predictor, then its performance or accuracy at a given data point (x_i, y_i) can be measured, for example, by the squared error loss: (not the only way to quantify error)

$$(h(x_i) - y_i)^2.$$

So, for the n data points above, we can check the performance of our prediction h and get a n real numbers:

$$(h(x_1) - y_1)^2, \quad (h(x_2) - y_2)^2, \quad \ldots, \quad (h(x_n) - y_n)^2.$$

Average error: What do we do with the n performance numbers?

$$(h(x_1) - y_1)^2, \quad (h(x_2) - y_2)^2, \quad \ldots, \quad (h(x_n) - y_n)^2.$$

A standard approach is to take the average of these numbers and define the sample error

$$L_S(h) = \frac{1}{n} \sum_{i=1}^{n} (h(x_i) - y_i)^2 .$$

We could have also taken the maximum of these numbers.

Quantifying accuracy: How good a predictor is at any point can be measured or quantified by penalizing the error $h(x_i) - y_i$. Squared error is one option. Other options are:

- $|h(x_i) - y_i|$
- $(h(x_i) - y_i)^p$
- $e^{|h(x_i) - y_i|}$

Training for regression: In most cases, training for regression is simply solving the following optimization problem: let $y_1, \ldots, y_n$ be real numbers. Then solve

$$\min_h \quad \frac{1}{n} \sum_{i=1}^{n} (h(x_i) - y_i)^2 .$$

Here the objective is to minimize average squared error loss over all possible maps from data x to y.

Some fundamental Questions

Given a supervised learning problem, how do we find a predictor? Well, you can always guess one, if that is possible for you. A more systematic approach is to solve the above training optimization problem.

What error criterion should I choose? You can choose any. But, people generally chosse squared error for regression. Squared error is popular because it is differentiable and leads to tractable mathematical equations. For classification, see discussion in the coming slides.

How do I solve the optimization problem encountered in training? We will, of course, discuss all that very soon. That is why there is a course on ML.

How do we know that our learned predictor will actually work well on unseen data? This is a really deep question. In general, training is not guaranteed to work. We will also discuss this topic soon.

Are there other ways to train so that that becomes successful? This is really asking the previous question in a different way. Again, more on this soon.

Example: Assume that we are given $n = 2$ pairs of data points.

y	x_1	x_2
0.1	1	2
0.2	2	2

- Here $x = (x_1, x_2) \in \mathbb{R}^2$ and $y \in \mathbb{R}$. So, $d = 2$.
- A predictor is a function of type $h : \mathbb{R}^2 \to \mathbb{R}$.
- For example, one possible predictor is

$$h(x_1, x_2) = \frac{x_1 + x_2}{10}.$$

- The average prediction error is

$$\frac{(h(1,2) - 0.1)^2 + (h(2,2) - 0.2)^2}{2} = \frac{(0.3 - 0.1)^2 + (0.4 - 0.2)^2}{2} = 0.04.$$

- Can you think of a predictor with lower prediction error?

Training for classification: Here instead of minimizing squared error loss, because y are discrete labels or numbers, a possible training process is

$$\min_h \quad \frac{\#\{i : h(x_i) \neq y_i\}}{n}.$$

Basically, count the number of times you mis-classify and divide by the sample size n.

Example: Assume that we are given $n = 3$ pairs of data points.

y	x_1	x_2
A	10	21
A	2	22
B	3	1

Suppose your predictor maps everything to the label A:

$$h(x_1, x_2) = A, \text{ for all } x_1, x_2.$$

Its accuracy is $\frac{1}{3}$.

Recall that we are given data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.

Training using a general loss function: Let

$$\ell(\tilde{y}, \hat{y}) \geq 0 \quad \text{(with equality if } \tilde{y} = \hat{y})$$

be a loss function such that given two values of variable y, $\tilde{y}$ and $\hat{y}$, returns a penalty $\ell(\tilde{y}, \hat{y})$. The more different $\tilde{y}$ and $\hat{y}$ are the larger the penalty $\ell(\tilde{y}, \hat{y})$. Training using a general loss function is solving this optimization problem

$$\min_h \quad \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i).$$

Basically, we find the function h that minimizes the average loss incurred over the training data.

Note that squared error loss and misclassification loss are special cases.

Squared Error Loss:

$$\ell(\tilde{y}, \hat{y}) = (\tilde{y} - \hat{y})^2.$$

So, if $\hat{y} = y_i$ and $\tilde{y} = h(x_i)$, then

$$\ell(h(x_i), y_i) = (h(x_i) - y_i)^2.$$

Mis-classification Error Loss:

$$\ell(\tilde{y}, \hat{y}) = \begin{cases} 1 & \text{if } \tilde{y} \neq \hat{y} \\ 0 & \text{if } \tilde{y} = \hat{y} \end{cases}$$

So, if $\hat{y} = y_i$ and $\tilde{y} = h(x_i)$, then

$$\ell(h(x_i), y_i) = \begin{cases} 1 & \text{if } h(x_i) \neq y_i \\ 0 & \text{if } h(x_i) = y_i. \end{cases}$$

The Output of Training and it's Use

Let h_s be the output of our training process. Mathematically,

$$h_s = \arg\min_h \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i).$$

Now that you have h_s, what will you do with it?

- **Diabetes Data**: You can sell your predictor to a clinic to help with predictions on future patients.
- **Digits Data**: You can sell it to USPS for help with automatic sorting based on zip codes. If USPS already have one they use, may be your predictor is better and you can convince them to use your predictor.
- **Image classification**: If you have an image classifier that is better than every thing else in the market, you can start a company and may be Google/Amazon will take over.

Does Training Work? Types of Errors

The biggest fact about ML is that training does NOT always work! But, what do we really mean by this question in the first place and in what sense training does not always work?

Recall that the output of the training process is a predictor h_s given by

$$h_s = \arg\min_h \frac{1}{n}\sum_{i=1}^{n} \ell(h(x_i), y_i).$$

The quantity $\frac{1}{n}\sum_{i=1}^{n} \ell(h(x_i), y_i)$ is called the **sample error**. We denote it by the symbol $L_S(h)$:

$$L_S(h) = \frac{1}{n}\sum_{i=1}^{n} \ell(h(x_i), y_i).$$

Suppose we are given m **new data points that we have never seen before**:

$$(x_{n+1}, y_{n+1}), (x_{n+2}, y_{n+2}), \ldots, (x_{n+m}, y_{n+m}).$$

Assuming that m is extremely large, possibly infinite, the **generalization error of the predictor** h_s is defined as

$$L_G(h_s) = \frac{1}{m} \sum_{j=1}^{m} \ell(h_s(x_{n+j}), y_{n+j}).$$

Technically, the correct definition is

$$L_G(h_s) = \lim_{m \to \infty} \frac{1}{m} \sum_{j=1}^{m} \ell(h_s(x_{n+j}), y_{n+j}).$$

But, we will pretend that m is some very large number. We will revisit this point when we look at mathematics of machine learning later in the course.

How to interpret the generalization error? The generalization error is the performance of the learned predictor h_s on dataset that it has not seen before. Intuitively, this is the true test of a predictor.

Quick Review of Definitions

Training Data: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$

Training: Training is minimizing sample error

$$\min_h L_S(h) = \min_h \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i)$$

$L_S(h)$ is called the sample error.

Output of training: The output of training is the minimizing function h_s:

$$h_s = \arg\min_h L_S(h) = \arg\min_h \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i)$$

Generalization Error: Performance of h_s on new data:
$(x_{n+1}, y_{n+1}), (x_{n+2}, y_{n+2}), \ldots, (x_{n+m}, y_{n+m})$ (m is large):

$$L_G(h_s) = \frac{1}{m} \sum_{j=1}^{m} \ell(h_s(x_{n+j}), y_{n+j}).$$

Average error: What do we do with the n performance numbers?

$$(h(x_1) - y_1)^2, \quad (h(x_2) - y_2)^2, \quad \ldots, \quad (h(x_n) - y_n)^2.$$

A standard approach is to take the average of these numbers and define the sample error

$$L_S(h) = \frac{1}{n} \sum_{i=1}^{n} (h(x_i) - y_i)^2 .$$

We could have also taken the maximum of these numbers.

Quantifying accuracy: How good a predictor is at any point can be measured or quantified by penalizing the error $h(x_i) - y_i$. Squared error is one option. Other options are:

- $|h(x_i) - y_i|$
- $(h(x_i) - y_i)^p$
- $e^{|h(x_i) - y_i|}$

Training for regression: In most cases, training for regression is simply solving the following optimization problem: let $y_1, \ldots, y_n$ be real numbers. Then solve

$$\min_h \quad \frac{1}{n} \sum_{i=1}^{n} (h(x_i) - y_i)^2 .$$

Here the objective is to minimize average squared error loss over all possible maps from data x to y.

When is the Training Process called Successful?

The training process is called successful if the generalization error of the predictor h_s is small.

How to interpret the generalization error? Recall that the generalization of the predictor h_s is

$$L_G(h_s) = \frac{1}{m} \sum_{j=1}^{m} \ell(h_s(x_{n+j}), y_{n+j}).$$

The generalization error is the performance of the learned predictor h_s on dataset that it has not seen. Intuitively, this is the true test of a predictor.

The most important fact about ML is that if we are not careful in the training process, the generalization error can be really poor.

Factors Affecting Successful Training

Successful Training = low generalization error on potentially infinitely large unseen data. What are the factors affecting successful training?

- **Sample size**: Can you learn from just one sample? What if there are two classes? **How large should be the training data?**
- **The type of functions h over which you search for optimal solution during training**: Consider the following predictor:

$$h_s(x) = \begin{cases} y_i, & \text{if } x = x_i \\ 0, & \text{otherwise.} \end{cases}$$

This **overfits** your training data, making perfect prediction on the training set. **How do we avoid functions like this or avoid overfitting in general?**

- **Independent and Identically Distributed (i.i.d.) Data**: If you want to sell your predictor, you have to be sure that it will be applied in future to data having similar $x - y$ relationship.

Sample Size

- **Too few samples won't work!**: If the number of samples are too small, we may not have enough information to understand the relationship between x and y.
- **How large should be the sample size?**: Intuitively, we need the training dataset size to be as large as possible? But, how to express this idea mathematically? There is a rich ML theory that can help us to quantify the training data size needed for successful training. **We will come back to this later in the course**.
- **Fixed sample size**: In most applications, the training data is something that is given to us. We do not necessarily have control over the size. We cannot ask people to collect more data for us. Again, there is ML theory that can tell you how well can you do for a given sample size.
- **Costly sample size**: Even if collecting data is under your control, the collection process may be quite costly (need to pay volunteers) or prohibitive (think cancer patients).

Class of Predictor Over Which we Search

- **Sample Error is Positive**: Since the loss function is positive,

$$L_S(h) = \frac{1}{n}\sum_{i=1}^{n} \ell(h(x_i), y_i) \geq 0.$$

- **A Bad Predictor**: What is the sample error for this predictor?

$$h_s(x) = \begin{cases} y_i, & \text{if } x = x_i \\ 0, & \text{otherwise.} \end{cases}$$

Obviously,

$$L_S(h_s) = 0.$$

- **Poor Generalization Error**: This predictor will perform poorly on new datasets because on every other point it selects a fixed value of zero. The only way this predictor is going to work is if exactly the same values of xs are observed, which is highly unlikely.

How do we avoid such a predictor? We will discuss this later in the course.

Independent and Identically Distributed Data

Let us think about the following situations. All of these may lead to poor generalization error.

- **Diabetes Data**: The training data came from old patients (age above 65). But, after the predictor is sold, the clinic starts to apply it to younger patients. The diagnosis might be different for patients of different profile.
- **Digits Data**: After your predictor is sold to USPS, some of the new letters arriving are written in a calligraphic manner. Your predictor may fail because it was not training on data of that type.
- **Image classification**: Suppose your predictor can count the number of objects in any image. But, when you sell it to a museum, they start applying it to modern art or paintings. And you trained your predictor using i-phone based images.

Intuitively, the generalization error will be small if the new data is similar to the training data. The right way to capture this notion of similarity is through the mathematical tools of probability and statistics (later in the course).