# The motivation of machine learning and mathematics model

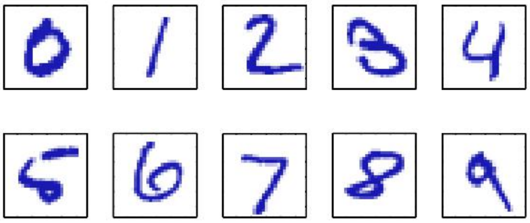
**Ken Wang**

In this class, I've tried to convey to you a broad a set of principles and tools that will be useful for doing many many things, and I am really confidence that this class things will really useful no matter what you will do after you finish this class. First, let's say a few more words about machine learning and mathematics model.

**Statistical learning(or machine learning)** refers to a vast set of tools for understanding data. These tools can be classified as supervised or unsupervised. Broadly speaking, **supervised statistical learning** involves building a statistical model for predicting, or estimating, an output based on one or more inputs. Problems of this nature occur in fields as diverse as business, medicine, astrophysics, and public policy. With **unsupervised statistical learning**, there are inputs but no supervising output; nevertheless we can learn relationships and structure from such data.

Example.1 Hand-Writting

Figure 1.1 Examples of hand-written digits taken from US zip codes.



(recognizing handwritten digits)

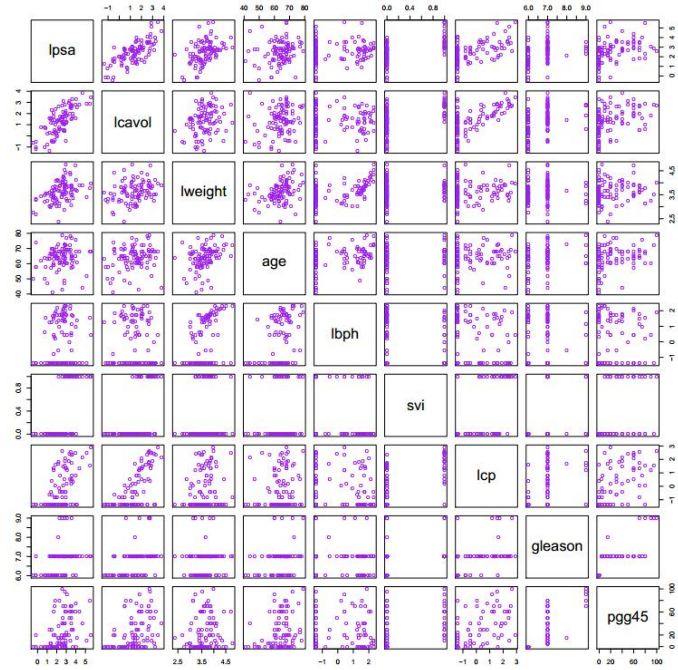
Consider the example of recognizing handwritten digits, each digit corresponds to a 28×28 pixel image and so can be represented by a vector x comprising 784 real numbers.

784 **features**

$$\text{N samples}\left[\begin{bmatrix} \mathbf{1} & \ldots\ldots & \ldots\ldots & 0 \\ . & \ldots\ldots & \ldots\ldots & . \\ . & \ldots\ldots & \ldots\ldots & . \\ 1 & \ldots\ldots & \ldots\ldots & 0 \end{bmatrix}\right. \xrightarrow{f} \begin{bmatrix} \mathbf{2} \\ \cdot \\ \cdot \\ \mathbf{5} \end{bmatrix}$$

The goal is to build a machine that will take such a vector x as **input** and that will produce the identity of the digit 0, . . . , 9 as the **output**. So statistical learning or machine learning, is try to use statistical methods to **training** a **model(learner)** to learn the functions. It is called "supervised" because of the presence of the outcome variable to guide the learning process. Because the output is categorical, so this is a **classification problem**

Example2. Prostat Cancer



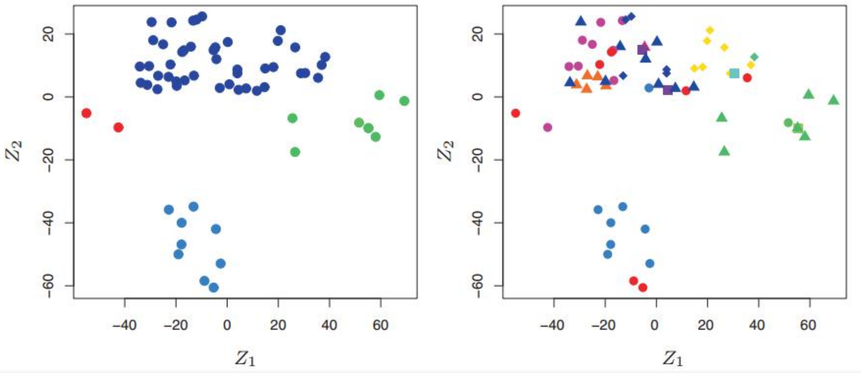
(predict the prostate specific antigen)

The data for this example come from a study by Stamey et al. (1989) that examined the correlation between the level of prostate specific antigen (PSA) and a number of clinical measures, in 97 men who were about to receive a radical prostatectomy.

The goal is to predict the log of PSA (lpsa) from a number of measurements including log cancer volume (lcavol), log prostate weight lweight, age, log of benign prostatic hyperplasia amount lbph, seminal vesicle invasion svi, log of capsular penetration lcp, Gleason score gleason, and percent of Gleason scores 4 or 5 pgg45. The figure is a scatterplot matrix of the variables. Some correlations with lpsa are evident, but a good predictive model is difficult to construct by eye. Besides, which **variable** that contribute directly to the PSA but not the results of PSA.

This is a supervised learning problem, known as a **regression problem**, because the outcome measurement is quantitative and continues.

Example 3. Gene Expression Profile



(The clustering analysis of different cell line)

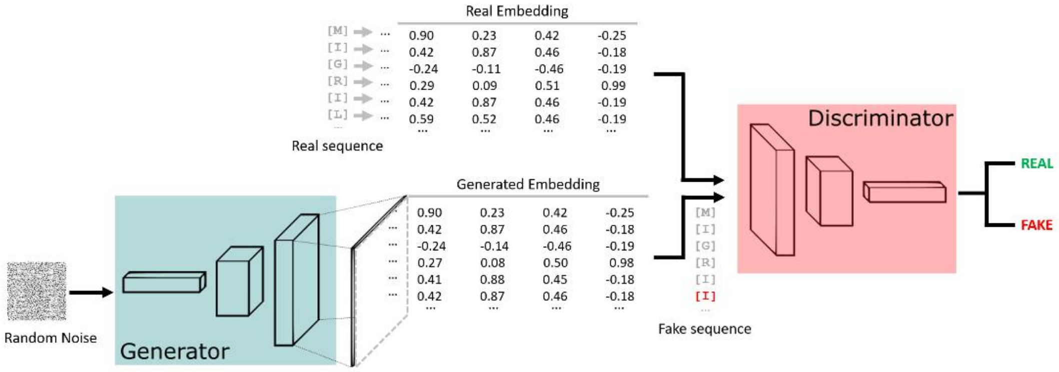
We consider the NCI60 data set, which consists of 6,830 gene expression measurements for each of 64 cancer cell lines. Instead of predicting a particular output variable, we are interested in determining whether there are groups, or clusters, among the cell lines based on their gene expression measurements.

But, We can perform PCA analysis, and select top 2 **principal components** of the data, which summarize the 6, 830 expression measurements for each cell line down to two numbers or dimensions. No matter the information may loss during dimension reduction process, it is now possible to visually examine the data for evidence of clustering.

For these questions with out label or outputs, we call it **unsupervised learning problem.**

Advanced Example. 1 2018iGEM

Team:Vilnius-Lithuania-OG **Best Model Awards Overgraduate**



| Real sequence | Real Embedding | | | |
|---|---|---|---|---|
| [M] → ... | 0.90 | 0.23 | 0.42 | -0.25 |
| [I] → ... | 0.42 | 0.87 | 0.46 | -0.18 |
| [G] → ... | -0.24 | -0.11 | -0.46 | -0.19 |
| [R] → ... | 0.29 | 0.09 | 0.51 | 0.99 |
| [I] → ... | 0.42 | 0.87 | 0.46 | -0.19 |
| [L] → ... | 0.59 | 0.52 | 0.46 | -0.19 |
| ... | ... | ... | ... | ... |

| Generated Embedding | | | | Fake sequence |
|---|---|---|---|---|
| 0.90 | 0.23 | 0.42 | -0.25 | [M] |
| 0.42 | 0.87 | 0.46 | -0.18 | [I] |
| -0.24 | -0.14 | -0.46 | -0.19 | [G] |
| 0.27 | 0.08 | 0.50 | 0.98 | [R] |
| 0.41 | 0.88 | 0.45 | -0.18 | [I] |
| 0.42 | 0.87 | 0.46 | -0.18 | [I] |
| ... | ... | ... | ... | ... |

Using GAN(generative adversarial networks) to creat a novel biological parts

They input a lot of real world protein sequences(409900 sequences) and used random noise to generate fake protein sequence to teach machine to design protein!

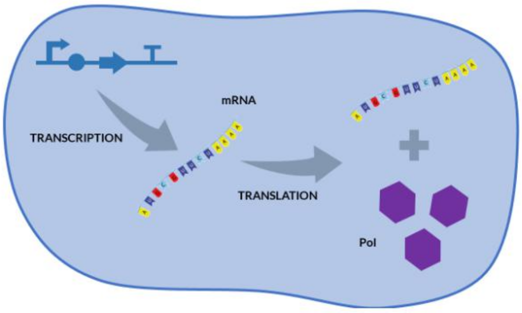
ProteinGAN - generative adversarial network architecture that they developed and optimized to generate de novo proteins. Once trained, it outputs a desirable amount of proteins with the confidence level of sequences belonging to a functional class it was trained for.

A Sub-neural network called discriminator learns how proteins look like by trying to distinguish between real and fake (generated) sequences. At the same time, another Sub-neural network called generator tries to fool it, by providing realistically looking generated protein sequences. Yet, the generator never actually sees any of the protein sequences.

Wiki Address: [http://2018.igem.org/Team:Vilnius-Lithuania-OG/ProteinGAN](http://2018.igem.org/Team:Vilnius-Lithuania-OG/ProteinGAN)

Advanced Example. 2 2018iGEM

Team:Valencia_UPV **Best Model Awards Undergraduate**



Constitutive expression model. Cell schema.

$$\xrightarrow{\mathrm{C_R}} \mathrm{mRNA}$$
$$\mathrm{mRNA} \xrightarrow{\mathrm{p}} \mathrm{mRNA} + \mathrm{PoI}$$
$$\mathrm{mRNA} \xrightarrow{\mathrm{d_m}} \emptyset$$
$$\mathrm{PoI} \xrightarrow{\mathrm{d_p}} \emptyset$$

Constitutive expression model. Biochemical reactions.

At the beginning of the design of any mathematical model, we have started from a cell scheme in which all the biochemical reactions happen. From the reactions, a set of equations is inferred. Each of these equations describe the temporal evolution of the main biochemical species in the cell (DNA, RNA, proteins and transcription factors), and depend on a set of parameters with a physical meaning.
Wiki Address: http://2018.igem.org/Team:Valencia_UPV/Model

## Basic concept

### Terminology

We begin by introducing a simple regression problem, which we shall use as a running example throughout this chapter to motivate a number of key concepts.
**regression** when we predict quantitative outputs
**classification** when we predict qualitative outputs
$X$ : input variable, its components can be accessed by subscripts $X_i$
$Y$ : output variable

$x_i$ : the ith sample or instance

$x^{(i)}$ : the ith variable.

$x_i^T = \left(x_i^{(1)}, x_i^{(2)}, x_i^{(3)} ... x_i^{(p)}\right)$

$$x^{(i)} = \left(x_1^{(i)}, x_2^{(i)}, ......, x_N^{(i)}\right)^T$$

$\mathbf{X}$ : N ×p matrix **X**

$\hat{Y}$ : Prodiction of Y based on X

Training Data: Learning Model Test Data: Prediction and evaluation

$T = \{(x_i, y_i)\}_{i \in \{1,2,...N)}$

### Simple Regression Problem

Given a vector of input X we predict out put Y via

$$\hat{Y} = \widehat{\beta}_0 + \sum_{j=1}^{p} X_j \widehat{\beta}_j$$

But we can write this formula into as an inner product

$$\widehat{Y} = X^T \widehat{\beta}$$

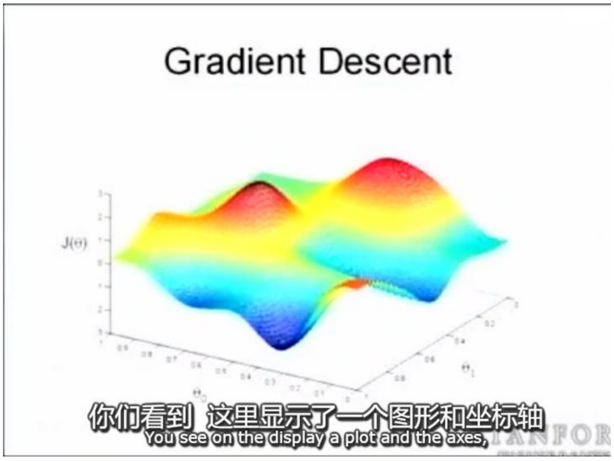
So how do we fit the linear model to a set of training data? First, we need to define **loss function**, or in statistics, **likelihood function**

$$RSS(\beta) = \sum_{i=1}^{N} \left( y_i - x_i^T \beta \right)^2$$

This function is quadratic function, which means its minimum is existence but not unique. So how to find its minimum?

### Gradient decent

Gradient decent is a search algorithm, we start from some initial value of my parameter vector beta(beta = 0), keep change beta to reduce our loss function.



(Gradient decent and local optimal)

How to choose the steepest direcation when you at mountain?

Gradient decent algorithm:

$$\beta := \beta - \alpha \frac{\partial loss}{\partial \beta}$$

For ith component of beta, we can take partial derivative respect to beta$_i$ , so we can get, for ith sample:

$$\frac{\partial loss}{\partial \beta^{(j)}} = \left( \widehat{y}_i - y_i \right) x^{(j)}$$

So here we has learning algorithm

Repeat till converge

$$\beta^{(i)} := \beta^{(i)} - \alpha \sum_{j=1}^{n} \left( \widehat{y}_j - y_j \right) x_j^{(i)}$$

Because the update function including overall training samples, we called it **batch gradient decent**.

### Linear algebra Perspective

In linear algebra perspective, the loss function can be re-writted into:

$$RSS(\beta) = \left( y - \mathbf{X}\beta \right)^T \left( y - \mathbf{X}\beta \right)$$

We can take derivative of beta of this equation, and let it equal to zero.

$$\mathbf{X}^T(y - \mathbf{X}\beta) = 0$$

This function, which we call it **normal function**, can give us the unique solution of beta, if X$^T$X is nonsingular(or inversible)

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

About matrix calculation and more details of gradient will left to next class.