Machine Learning

- Design an algorithm is not easy
- What if a machine can learn...
Fish Packing Plant wants to automate the process of sorting incoming fishes (Salmon / Sea Bass) on a belt according to species.

How to design a system?

Rule-base system is commonly used

Example: For a Fish
- If Length > 10 and Fin Area > 10, Sea Bass
- If Weight > 4.3 or Length < 4, Salmon

Factor of a rule:
- Characteristic (Feature) Length, Weight, Color, Shape of fin / head,
- Quantification (Threshold) Fin area > 10, Sea Bass Fin area < 10, Salmon
Difficult to determine a rule manually, even for an expert
- How to define the rules (Feature and Threshold)?
- E.g. If Weight > 4.3 or Length < 4, Salmon

Machine Learning can help
- You do not know how but implement an algorithm which can learn from data

Machine Learning: Fish Packing Plant Example

Salmon / Sea Bass

- Preprocessing (Isolate Fish, reduce noise…)
- Feature Extraction (Take Measurement)
- Classification

Fish
- Sensing (camera)
- Image
- Refined Image
- Input Features

Output
Class (Salmon / Sea Bass)
**Sensing**

- **Digitize** the object to the format which can be handled by machines

**Example**

- **Type of Device**
  - Camera? Depth Camera? Infra-red?
  - Ultrasound? Movement Sense? Combination?

- **Setting of Device**
  - Number? Angle? Overlap shooting range?

- **Background**
  - Lighting? Background simplicity?

---

**Preprocessing**

- **Refine** the data

**Example**

- Lighting conditions
- Position of fish
- Angle of fish
- Noise
- Blurriness
- Segmentation (remove object from background)
**Feature Extraction**
- Decide which **information** is able to distinguish classes
- Example
  - Length, width, weight, number and shape of fins, tail shape, etc.
- Rely on technical background and common sense
  - Experts may help

**Decision Making**
- Decision Type:
  - **Class** (Classification)
  - **Value** (Regression, Value Prediction)
  - **Rank** (Ranking)
  - **Action** (Reinforcement Learning)
  - **Region** (Segmentation)
- Many machine learning techniques are available
**Sensing**
Assume a fish is put on a belt and a single camera is installed to take a photo on each fish.

**Preprocessing**
Remove the blueness and noise.

---

**Feature Extraction**

- The expert (e.g., Fisherman) suggests salmon is usually shorter than sea bass.
- Length is chosen (as a feature) as a decision criterion.
15 is selected as the threshold

Although sea bass is longer in general, there are many exceptions

The experts “may be” wrong!

How about other features?

E.g. lightness

---

Try another feature “Lightness”

5.5 is selected as the threshold

“lightness” is better than “length”
Besides accuracy, “costs of different errors” can be considered

**Case 1: Company’s view**
- Salmon is more expensive than sea bass. Selling Salmon with the price of sea bass will be a loss
  - If salmon is classified as sea bass: **HIGH cost**
  - If sea bass is classified as salmon: **LOW cost**

**Case 2: Customer’s view**
- Customers who buy salmon will be upset if they get sea bass; Customers who buy sea bass will not be upset if they get the more expensive salmon
  - If salmon is classified as sea bass: **LOW cost**
  - If sea bass is classified as salmon: **HIGH cost**
**Case 2:**
Customer’s view

- **LOW** cost
  Salmon is classified as sea bass
- **HIGH** cost
  Sea bass is classified as salmon
- Avoid classifying sea bass wrongly by scarifying salmon
Dr. Patrick Chan @ SCUT

Machine Learning: Fish Packing Plant Example

Multiple Features

- Only ONE feature may not be good enough
- **More features** should be considered
- Two features: Lightness \((x_1)\), Width \((x_2)\)
- A fish is represented by a point in a 2D feature space:

\[
x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}
\]

Classifier

- A **decision boundary** can be drawn to divide the feature space into two regions
- Is it a linear classifier too simple?

What is this unseen fish?
Will other classifiers be better?
- More complex classifier
- Perfectly classify training samples
- Ultimate objective is to classify unseen samples correctly
- Can it be generalized to unseen sample?

Tradeoff between accuracy of training samples and complexity
- Look more reasonable
  - Not too complex
  - Good in classifying the training samples
Supervised Learning
Correct / Wrong

Unsupervised Learning
No ground truth

Reinforcement Learning
Learn from reward

Supervised Learning
- Ground truth (desired output) is provided
- A sample \((x, y)\)
  - \(x\): a feature vector
  - \(y\): a desired output (e.g. label, value, ...)
- Learn the mapping between \(x\) and \(y\)
- Predict \(y\) for an unseen \(x\)
- Error can be measured explicitly
**Classification**

- $y$ is a *label* of the sample
- E.g. $x = (\text{Length, Weight})$
  
  $y = \text{Seabass or Salmon}$

**Regression**

- $y$ is a *real number*
- E.g. $x = (\text{Length})$
  
  $y = \text{Price of a fish}$
Supervised Learning

Example

Classification

- Class (Classification)

Object Detection

- For each bounding box
  - Size and Coordination (Regression)
  - Class (Classification)

Classification + Localization

A bounding box
- Size and Coordination (Regression)
- Class (Classification)

Instance Segmentation

- For each bounding box
  - Size and Coordination (Regression)
  - Class (Classification)
  - Which pixel is background? (Classification)

Agenda

- Supervised Learning
  - Regression
    - Least Mean Square
  - Classification
    - Probability
Regression

- Given 15 fishes: weight and prices
- Objective: Predict the price of a fish

<table>
<thead>
<tr>
<th>Weight (kg)</th>
<th>Price ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2</td>
<td>20</td>
</tr>
<tr>
<td>2.6</td>
<td>31</td>
</tr>
<tr>
<td>1.2</td>
<td>16.5</td>
</tr>
<tr>
<td>0.7</td>
<td>10</td>
</tr>
</tbody>
</table>

The $i^{th}$ training sample: $(x^{(i)}, y^{(i)})$

- $x^{(i)} = [x_1^{(i)}, x_2^{(i)}, \ldots, x_d^{(i)}]^T \in X$: Feature vector
  - $x_j^{(i)}$: the jth feature
  - $X$: the input space
- $y^{(i)} \in Y$: Target Value
  - $Y$: the output space

Training set: $\{(x^{(i)}, y^{(i)}) | i = 1 \ldots n\}$
- $n$ is the number of training samples
Regression

- Train a function $h_{\theta}(x)$ to predict $y$
  - $\theta$ is the parameter vectors of the model
  - Different parameters yields different predictions (different $h$)
  - Example: weight
  - $h_{\theta}: X \rightarrow Y$, mapping from $X$ to $Y$
  - $h_{\theta}$ is called a predictor or hypothesis

- Objective: Build a “good” $h_{\theta}$
  - What does “good” mean?

Objective Function

- Objective: the predicted value on a training sample closer to the real one
  - Smaller difference between $h_{\theta}(x^{(i)})$ and $y^{(i)}$

- Cost function (objective function)
  - May contains other terms besides Error
  - Mean Square Error is a classic measure

\[
J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} (h_{\theta}(x^{(i)}) - y^{(i)})^2
\]

- Error is a distance measure
- Square avoids the cancellation of positive and negative error
**LMS Algorithm**

- **Least Mean Squares (LMS)** aims to minimize $J(\theta)$ by adjusting $\theta$

\[ J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} (h_{\theta}(x^{(i)}) - y^{(i)})^2 \]

- ML is closely related to Optimization problem
  - Usually, the optimization is quite complicated
    - Since each parameter is a variable
    - Iterative method is used

---

**LMS Algorithm**

**Gradient Descent**

- When $h_{\theta}$ is differentiable, gradient descent can be used to minimize $J(\theta)$

\[ J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} (h_{\theta}(x^{(i)}) - y^{(i)})^2 \]

- Influence on $J(\theta)$ by changing the parameter ($\theta$) slightly

\[ \theta^{(t+1)} = \theta^{(t)} - \alpha \frac{\partial J(\theta^{(t)})}{\partial \theta} \]

  - $\alpha$ : the learning rate
  - $\theta^{(t)}$ : the parameter at the time $t$
LMS Algorithm

Gradient Descent

◆ Algorithm
  ■ Start with an arbitrarily chosen weight $\theta^{(1)}$
  ■ Let $t = 0$
  ■ Loop
    ■ $t = t + 1$
    ■ Compute gradient vector $\nabla J(\theta^{(t)}) / \partial \theta$
    ■ Next value $\theta^{(t+1)}$ determined by moving some distance from $\theta^{(t)}$ in the direction of the steepest descent
      $$\theta^{(t+1)} = \theta^{(t)} - \alpha \frac{\partial}{\partial \theta} J(\theta^{(t)})$$
    ■ i.e., along the negative of the gradient
  ■ Until Finish Training

Recall, $\theta = [\theta_1, \theta_2, ..., \theta_m]$

Updated Rule for the $j^{th}$ parameter

$$\theta_j^{(t+1)} = \theta_j^{(t)} - \alpha \frac{\partial}{\partial \theta_j} J(\theta_j^{(t)})$$

All parameters should be updated at the same time
### LMS Algorithm

#### Gradient Descent

- **How to calculate** \( \frac{\partial}{\partial \theta_j} J(\theta) \)?

\[
J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)})^2
\]

\[
\frac{\partial}{\partial \theta_j} J(\theta) = \frac{\partial}{\partial \theta_j} \frac{1}{2n} \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)})^2
\]

\[
= \frac{1}{2n} \sum_{i=1}^{n} \frac{\partial}{\partial \theta_j} (h_\theta(x^{(i)}) - y^{(i)})^2
\]

\[
= \frac{1}{2n} \sum_{i=1}^{n} 2(h_\theta(x^{(i)}) - y^{(i)}) \frac{\partial}{\partial \theta_j} (h_\theta(x^{(i)}) - y^{(i)})
\]

\[
= \frac{1}{2n} \sum_{i=1}^{n} 2(h_\theta(x^{(i)}) - y^{(i)}) \frac{\partial h_\theta(x^{(i)})}{\partial \theta_j}
\]

**Depend on a model**

### Example: Linear Function

\[ h_\theta(x) = \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_d x_d \]

\[
= \sum_{i=1}^{d} \theta_i x_i
\]

\[
\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} 2(h_\theta(x^{(i)}) - y^{(i)}) \frac{\partial h_\theta(x^{(i)})}{\partial \theta_j}
\]

\[
= \frac{1}{2n} \sum_{i=1}^{n} 2(h_\theta(x^{(i)}) - y^{(i)}) \frac{\partial}{\partial \theta_j} \left( \sum_{k=1}^{d} \theta_k x_k^{(i)} \right)
\]

\[
= \frac{1}{n} \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}
\]
LMS Algorithm

Gradient Descent

Related Issues:

- **Size of Learning Rate** ($\alpha$)
  - Too small, convergence is needlessly slow
  - Too large, the correction process will overshoot and cannot even diverge

- **Sub-optimal Solution**
  - Trapped by local minimum

- We will study Gradient Descent again in Artificial Neural Network

Classification

- **Objective**: output a class based on the features of a sample

Illustration: Hanne Moller, moliaren.dk
Peter went to body check to see if he is ok
\[ y = (\text{ill, healthy}) \]

According to the previous records, the doctor concluded
- 85% of people was healthy
  \[ P(y = \text{healthy}) = 0.85 \]
- 15% of people was ill
  \[ P(y = \text{ill}) = 0.15 \]
- Therefore, Peter was healthy
  \[ P(y = \text{healthy}) > P(y = \text{ill}) \]

Should Peter be satisfied with this diagnosis?
- This decision is based on Prior Probability \( P(y) \)

Physical condition of persons should be considered
- Quantify the characteristics (features), denoted by \( x \)
- E.g. red blood cell #, white blood cell #, temperature

Assume only “white blood cell #” is measured
Assume the white blood cell # \((x)\) of Peter is 2

A probability density function (pdf) of persons is considered

The Doctor said
- \(p(x=2 \mid \text{ill}) = 0.67\)
- \(p(x=2 \mid \text{healthy}) = 0.05\)
- Therefore, Peter is ill

Should we be satisfied?
- This decision is based on Likelihood \(p(x \mid y)\)

Using Prior Probability \((P(y))\) or Likelihood \((p(x \mid y))\) is not suitable

Posterior Probability is a better choice
\(P(y \mid x)\) : given \(x\), the probability of \(y\)

Bayes Decision Rule (Bayes Classifier)
- When \(P(y_1 \mid x) > P(y_2 \mid x)\), \(x\) is \(y_1\)
- When \(P(y_2 \mid x) > P(y_1 \mid x)\), \(x\) is \(y_2\)
- When \(P(y_1 \mid x) = P(y_2 \mid x)\), no decision

How to obtain \(P(y \mid x)\)?
- Obtaining from data is difficult as \(x\) is usually a continuous value
Bayes Formula

\[ P(y|x) = \frac{p(x|y)P(y)}{p(x)} \]

- Likelihood and prior probability may be estimated by using a dataset (*Discuss it later in the lecture*)
- How about evidence \( p(x) \)?

Bayes Decision Rule

- \( p(x) \) is difficult to be obtained relatively
  - \( p(x) \) contain all kinds of samples, which is more complicated than \( p(x|y) \)
  - It can be neglected in decision making

\( x \) is classified as \( y_1 \) if

\[ p(y_1|x) > p(y_2|x) \]

\[ \frac{p(x|y_1)P(y_1)}{p(x)} > \frac{p(x|y_2)P(y_2)}{p(x)} \]

\[ p(x|y_1)P(y_1) > p(x|y_2)P(y_2) \]
Given:  
\[ p(x=2 \mid \text{ill}) = 0.67 \quad p(x=2 \mid \text{healthy}) = 0.05 \]
\[ P(\text{ill}) = 0.15 \quad P(\text{healthy}) = 0.85 \]

Recall, Bayes Decision Rule:
- Decide \( y_1 \) if \( P(y_1 \mid x) > P(y_2 \mid x) \)
- Decide \( y_2 \) if \( P(y_2 \mid x) > P(y_1 \mid x) \)

\[
P(\text{healthy} \mid x = 2) \propto p(x=2 \mid \text{healthy}) \times P(\text{healthy})
= 0.05 \times 0.85 = 0.0425
\]

\[
P(\text{ill} \mid x = 2) \propto p(x=2 \mid \text{ill}) \times P(\text{ill})
= 0.67 \times 0.15 = 0.1005
\]

* Note that if \( p(x) \) is considered, then \( P(y_1 \mid x) + P(y_2 \mid x) = 1 \).

0.1005 > 0.0425, therefore, Peter is ill.
Error of Bayes Decision Rule

- True

<table>
<thead>
<tr>
<th>Predict: ( y_1 )</th>
<th>Predict: ( y_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct: ( y_1 )</td>
<td>Correct: ( y_2 )</td>
</tr>
<tr>
<td>Wrong: ( y_1 )</td>
<td>Wrong: ( y_2 )</td>
</tr>
</tbody>
</table>

Samples is healthy when prediction is ill
Samples is ill when prediction is healthy

\[
p(x|\text{healthy})P(\text{healthy})
\]

\[
p(x|\text{ill})P(\text{ill})
\]

\[
\{ x \mid x \text{ is classified as } y_1 \}
\]

\[
\{ x \mid x \text{ is classified as } y_2 \}
\]

\[
\{ x \mid x \text{ belongs to } y_1 \}
\]

\[
\{ x \mid x \text{ belongs to } y_2 \}
\]

Prediction

True

\[
P(\text{error}) = P(x \in \overline{X}_2, y_1) + P(x \in \overline{X}_1, y_2)
\]

\[
= P(x \in \overline{X}_2|y_1)P(y_1) + P(x \in \overline{X}_2|y_2)P(y_2)
\]

\[
= \int p(x|y_1)P(y_1)dx + \int p(x|y_2)P(y_2)dx
\]
Classification: Bayes Decision Rule

Error

- $\theta_b$ or $\theta'$ is better?
  - Error of $\theta_b < \theta'$
  - $\theta_b$ is better

- Is any boundary better than Bayes Decision Rule ($\theta_b$)?
  - Bayes Rule is optimal (minimal classification error)

- Error = Bayes Error + Added Error

**Bayes Error**
Error of Bayes Rule
- Cannot be reduced
- Depend on the input space and application

**Added Error**
Additional error made by other classifiers
- Can be reduced by selecting better parameters
Extension to Multi-Class

- Extend to multi-class problem \((c\) classes\)
  \[ y = (y_1, y_2, \ldots, y_c) \]

- Bayes Decision Rule
  - \(x\) is \(y_i\) if \(P(y_i|x)\) is maximum for \(i = 1 \ldots c\)

- Error for Bayes Decision Rule
  \[ P(error \mid x) = 1 - \max[ P(y_1|x), P(y_2|x), \ldots, P(y_c|x) ] \]

Three-class example

- Bayes Decision Rule
  - \(x\) is \(y_i\) if \(P(y_i|x)\) is max for \(i = 1 \ldots 3\)

\[ P(y_1 \mid x) \]
\[ P(y_2 \mid x) \]
\[ P(y_3 \mid x) \]
Error of Bayes Decision Rule:
\[ P(\text{error} \mid x) = 1 - \max[P(y_1 \mid x), P(y_2 \mid x), P(y_3 \mid x)] \]

For example, in the **green region**, (\(x\) is classified as \(y_2\) based on Bayes Rule)
\[
P(\text{error} \mid x) = P(y_1 \mid x) + P(y_3 \mid x) \quad * \text{Posterior probability (figures are not)}
\]
\[
= 1 - P(y_2 \mid x)
\]
\[
= 1 - \max_{i=1,2,3} P(y_i \mid x) \quad * \text{Must not be } P(y_2 \mid x)
\]

Therefore,
\[
P(\text{error} \mid x) = 1 - \max[P(y_1 \mid x), P(y_2 \mid x), P(y_3 \mid x)]
\]
Learning Type

◆ How to learn $P(x \mid y_i)$ and $P(y_i)$?
  
  **Parametric Methods** (Briefly Introduce here)
  
  ▶ Model-based Method *Assume form of sample distribution (pdf) is known*
  
  ▶ Estimate parameters of the distribution
  
  ▶ Bias (Very Good if the assumption is correct)

  **Non-Parametric Methods** (Learn in Next Lecture)
  
  ▶ Model Free Method *No assumption on pdf*
  
  ▶ A proper form for discriminant function is assumed
  
  ▶ Usually sub-optimal, but good results generally

---

**Parametric Methods**

**Normal Distribution**

◆ Assume samples in each class follow normal distribution (Gaussian distribution),

$$D \sim N(\mu, \sigma^2)$$

◆ Normal Density in $d$ dimensions is:

$$p(x) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp \left[ -\frac{1}{2} (x - \mu)^t \Sigma^{-1} (x - \mu) \right]$$

where:

- $x = (x_1, x_2, ..., x_d)^t$
- $\mu = (\mu_1, \mu_2, ..., \mu_d)^t$ : mean vector
- $\Sigma = d \times d$ : covariance matrix
- $|\Sigma|$ and $\Sigma^{-1}$ : determinant and inverse respectively
- $t$ : transpose
Assume the covariance matrices are the identity matrix, the distributions are spherical.

Assume each class has the same covariance matrix, the distributions are in ellipse sharp.
Parametric Methods
Normal Distribution \((\Sigma_i = \sigma^2 I)\)

- The covariance matrix can be anything

- Multi-class problem
  - Even with small number of classes, the shapes of the boundary regions is complex