# Machine Learning

- What If a Program could Program Itself ?

Abdus Salam Azad

# Machine Learning

- A way to automatically “write” a “program” based on input and outputs

    - “write” => “Learn”
    - “program” => can be a python program, a neural network, etc…

- In this lecture would simply touch a few basic concepts…

# Machine Learning

- In traditional programming, we are given a problem statement and a few input-output examples
  - We “figure out” the “logic” manually
  - and the we **manually** write the program

- In the most typical setting of ML:

  - we are given a set of inputs and outputs

  - we need to a program that automatically learns another “program” such that
    - it automatically “figures out” the “logic” of the program
    - given the example inputs can generate similar outputs
    - given unseen input, can generate “reasonable” output

# Machine Learning Example - Classification

When should I play Tennis ????

| Day | Outlook | Temperature | Humidity | Wind | PlayTenni |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

# Machine Learning Example - Classification

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

```
if Outlook == "Overcast":
    PlayTennis=Yes!!
```

# Classification - What if it's Sunny or Rainy ?

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| ~~D3~~ | ~~Overcast~~ | ~~Hot~~ | ~~High~~ | ~~Weak~~ | ~~Yes~~ |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| ~~D7~~ | ~~Overcast~~ | ~~Cool~~ | ~~Normal~~ | ~~Strong~~ | ~~Yes~~ |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| ~~D12~~ | ~~Overcast~~ | ~~Mild~~ | ~~High~~ | ~~Strong~~ | ~~Yes~~ |
| ~~D13~~ | ~~Overcast~~ | ~~Hot~~ | ~~Normal~~ | ~~Weak~~ | ~~Yes~~ |
| D14 | Rain | Mild | High | Strong | No |

# Observe the “Sunny” days

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |

# Observe the “Sunny” days

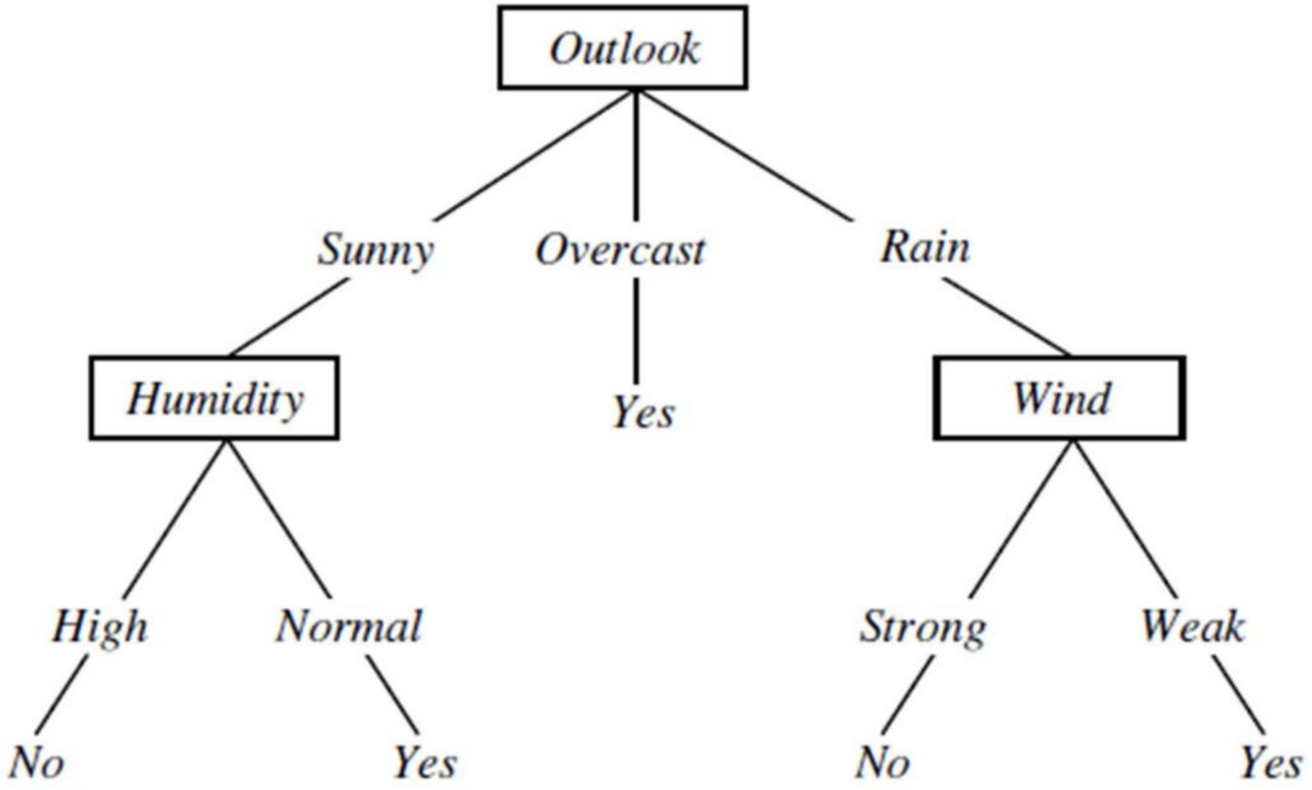
| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |

```
if Outlook == “overcast”:
      PlayTennis=Yes
elif Outlook == “Sunny” and Humidity == “High”:
      PlayTennis=No
elif Outlook == “Sunny” and Humidity == “Normal”:
      PlayTennis=Yes
```

# Observe the days with “Rain”

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D14 | Rain | Mild | High | Strong | No |

# Decision Trees: Learned a "program" in the form of a tree



# Learning Decision Trees: ID3

ID3(*Examples*, *Target_attribute*, *Attributes*)

*Examples are the training examples. Target_attribute is the attribute whose value is to be predicted by the tree. Attributes is a list of other attributes that may be tested by the learned decision tree. Returns a decision tree that correctly classifies the given Examples.*

- Create a *Root* node for the tree
- If all *Examples* are positive, Return the single-node tree *Root*, with label = +
- If all *Examples* are negative, Return the single-node tree *Root*, with label = −
- If *Attributes* is empty, Return the single-node tree *Root*, with label = most common value of *Target_attribute* in *Examples*
- Otherwise Begin
  - $A \leftarrow$ the attribute from *Attributes* that best* classifies *Examples*
  - The decision attribute for *Root* $\leftarrow A$
  - For each possible value, $v_i$, of $A$,
    - Add a new tree branch below *Root*, corresponding to the test $A = v_i$
    - Let $Examples_{v_i}$ be the subset of *Examples* that have value $v_i$ for $A$
    - If $Examples_{v_i}$ is empty
      - Then below this new branch add a leaf node with label = most common value of *Target_attribute* in *Examples*
      - Else below this new branch add the subtree
        ID3($Examples_{v_i}$, *Target_attribute*, $Attributes - \{A\}$))
- End
- Return *Root*

# Choosing The Best Attribute

- There are several different types of criteria
- We will briefly discuss one of them: Information Gain
- First we need to define a metric, Entropy

- $S$ is a sample of training examples
- $p_{\oplus}$ is the proportion of positive examples in $S$
- $p_{\ominus}$ is the proportion of negative examples in $S$
- Entropy measures the impurity of $S$

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

- Entropy Is **Zero (Minimum)**, When all the examples belong to **same class**!!!
- Entropy Is **1(Maximum)**, When there are **equal** number of positive and negative examples!

# Entropy Example

To illustrate, suppose $S$ is a collection of 14 examples of some boolean concept, including 9 positive and 5 negative examples (we adopt the notation $[9+, 5-]$ to summarize such a sample of data). Then the entropy of $S$ relative to this boolean classification is

$$\begin{aligned} Entropy([9+, 5-]) &= -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) \\ &= 0.940 \end{aligned} \tag{3.2}$$

$$Entropy(S) \equiv \sum_{i=1}^{c} -p_i \log_2 p_i$$

# Information Gain

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$\begin{aligned}
Values(Wind) &= Weak, Strong \\
S &= [9+, 5-] \\
S_{Weak} &\leftarrow [6+, 2-] \\
S_{Strong} &\leftarrow [3+, 3-] \\
Gain(S, Wind) &= Entropy(S) - \sum_{v \in \{Weak, Strong\}} \frac{|S_v|}{|S|} Entropy(S_v) \\
&= Entropy(S) - (8/14)Entropy(S_{Weak}) \\
&\quad - (6/14)Entropy(S_{Strong}) \\
&= 0.940 - (8/14)0.811 - (6/14)1.00 \\
&= 0.048
\end{aligned}$$

- Gain “estimates” reduction of entropy
- We select the attribute with Maximum Gain

# Classification Terminology

**Feature Vector**

Class Label

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

# Classification - A bit more formally

Given a set, S of (x,y) pairs, find a function f(x) that predicts correct y values

- x: feature vector
- y: label
- f(x): classifier
- S: Dataset

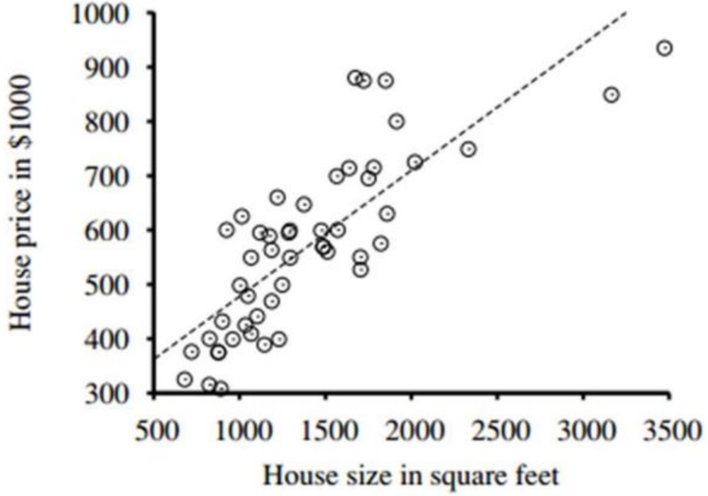
# Machine Learning: Regression

| House Area (sq. ft.) | Price (thousands) |
|---|---|
| 1656 | 215 |
| 896 | 105 |
| 1329 | 172 |
| ... | ... |

- Given a set S of (x,y) pairs, find a function f(x) that returns good y values

# Linear Regression

- "fits" a straight line
- The learned model is a "straight line"




```
def line(x):
     return mx + c
```

The training algorithm learns the weight "m" and bias "c"

find a "m" and "c" such that sqrt((y-f(x))^2) is minimum.
=> "root mean squared error"

# Univariate Linear Regression

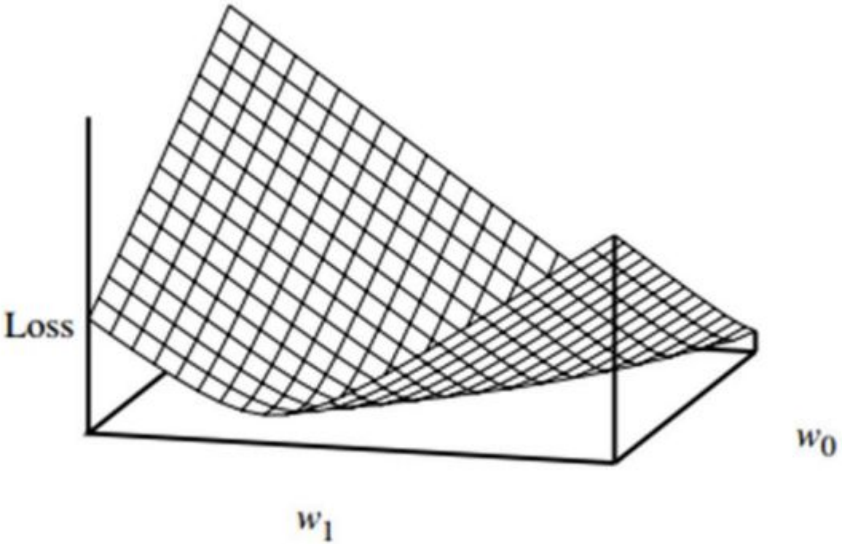
- A univariate linear function (a straight line) with input x and output y has the form
  y = w1*x+ w0,
  - w0 and w1 are real-valued “weights” (c and m respectively from previous slide)

- Weight Vector: w = [w0, w1]

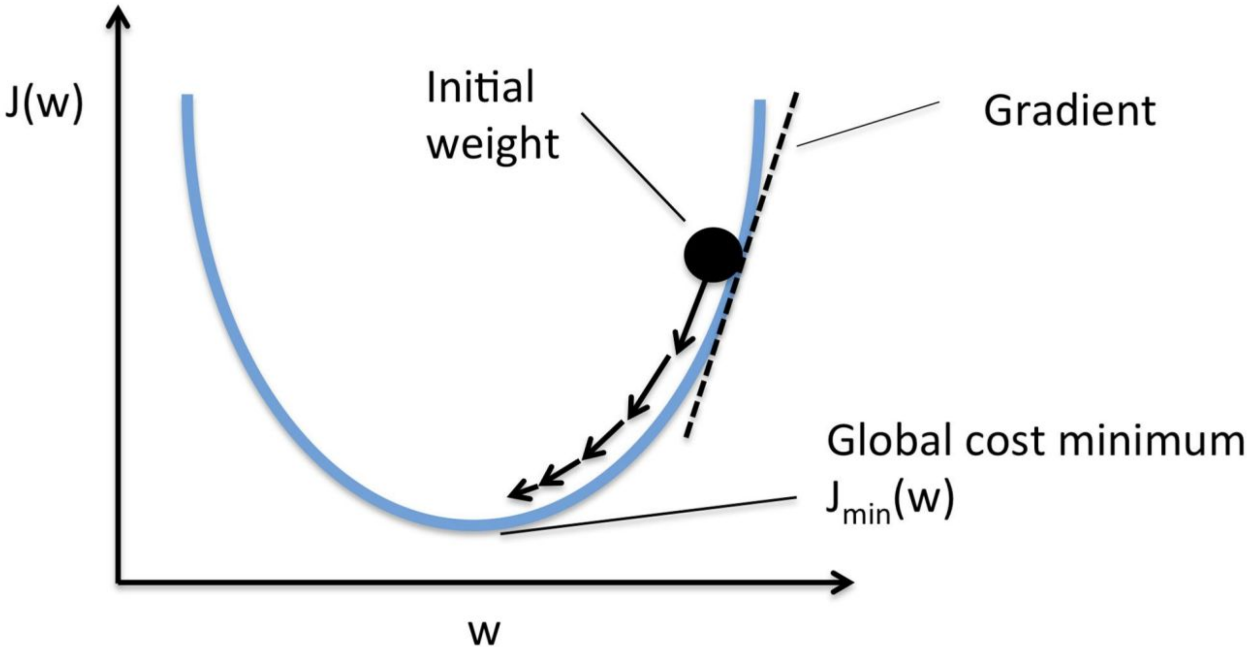
- Find the weight vector w` which minimizes the means squared error i.e., loss

$$\sum_{j=1}^{N}(y_j - (w_1 x_j + w_0))^2$$

# Weight Space and Minimization



# Gradient Descent



# Gradient Descent

$\mathbf{w} \leftarrow$ any point in the parameter space

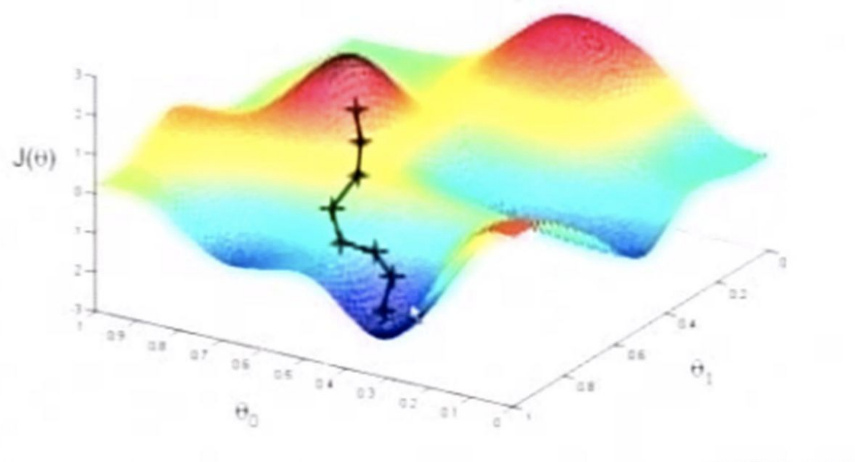
**loop** until convergence **do**

  **for each** $w_i$ **in** $\mathbf{w}$ **do**

$$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} Loss(\mathbf{w})$$

alpha is the learning rate

# Gradient Descent



# Food for Thought

- How to evaluate how good your learned model is?
  - Train and Test set

- What if the Training Data is Noisy ?
  - Problems of overfitting

# Broad Types of Machine Learning

- Supervised Learning
  - Classification
  - Regression

- Unsupervised Learning
  - Clustering
  - Embeddings

- Reinforcement Learning
  - e.g., Learning to solve games, e.g., Alpha Go


MAY THE FORCE
BE WITH YOU.™

# Acknowledgement of Resources

- Decision Tree Examples taken from Tom MItchell’s book
- Regression Example: Russel & Norvig – Chapter 18: 18.6.1,18.6.2
- Closed Form Solution for Linear Regression
    - https://towardsdatascience.com/normal-equation-in-python-the-closed-form-solution-for-linear-regression-13df33f9ad71