Machine Learning: "field of study that gives computers the ability to learn without being explicitly programmed." (Arthur Samuel, 1959)

Well-posed Learning Problem: "a computer program is said to LEARN from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E." (Tom Mitchell, 1998)

Machine Learning algorithms:

- Supervised learning
- Unsupervised learning
- Others: reinforcement learning, recommender systems.

Supervised Learning

- Regression: predict continuous valued output.
- Classification: predict discrete valued output.

Unsupervised Learning

- Cocktail party problem algorithm [Octave]:

```
[W,s,v] = svd((repmat(sum(x.*x,1),size(x,1),1).*x)*x');
```

- `svd`: Singular Value Decomposition
- `repmat`: repeat matrix

Linear Regression

Model representation

- Training set of housing prices (Portland, OR)
    - x: size in squared feet (input variable / feature)
    - y: price in thou. of US dollars (output variable / target)
    - m: number of training examples
    - (x, y): one training example
    - $(x^{(i)}, y^{(i)})$: i-th training example

  Training set -> Learning algorithm -> h (hypothesis)

    - h: maps from x to y.

  $h_\theta(x) = \theta_0 + \theta_1 x$

    - Shorthand: $h(x)$
    - Univariate linear regression.

Cost function

Hypothesis: $h(x) = \theta_0 + \theta_1 x$

- θ_i's: parameters
    - how to choose θ_i's?

- Optimization problem:
$$\underset{\theta_0,\theta_1}{min}\ J(\theta_0, \theta_1) := \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$
  - Squared error function, or mean squared error.
    - "The mean is halved as a convenience for the computation of the gradient descent, as the derivative term of the square function will cancel out the $\frac{1}{2}$ term."

Gradient Descent

Have some function $J(\theta_0, \theta_1)$ we want to minimize.
 - Start with some θ_0, θ_1;
 - Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ until we hopefully end up at a minimum.

- Algorithm:
  repeat until convergence{
  $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$ (for $j = 0$ and $j = 1$)
  }
- Correct: simultaneous update

`temp0` $:= \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$
`temp1` $:= \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$
$\theta_0 :=$ `temp0`
$\theta_1 :=$ `temp1`

- α: learning rate
- Applying Gradient Descent to Linear Regression

$$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_j} \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

- for $j = 0$: $\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)$
- for $j = 1$: $\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) \cdot x^{(i)}$

Matrices and Vectors

Matrix: rectangular array of numbers.

$$A = \begin{bmatrix} 1402 & 191 \\ 1371 & 821 \\ 949 & 1437 \\ 147 & 1448 \end{bmatrix}$$

- Dimension of matrix: number of rows x number of columns.

Matrix elements: entries of the matrix.

- $A_{ij} =$ the entry in i-th row, j-th column.

Vector: an $n \times 1$ (or $1 \times n$) matrix.

Vector elements:

- $y_i =$ the i-th element.

Matrix Addition

$$\begin{bmatrix} 1 & 0 \\ 2 & 5 \\ 3 & 1 \end{bmatrix} + \begin{bmatrix} 4 & 0.5 \\ 2 & 5 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 5 & 0.5 \\ 4 & 10 \\ 3 & 2 \end{bmatrix}$$

- The matrices must have the same dimensions!
- We add the elements with the same coordinates.

Scalar Multiplication

$$3 \cdot \begin{bmatrix} 1 & 0 \\ 2 & 5 \end{bmatrix} = \begin{bmatrix} 3 & 0 \\ 6 & 15 \end{bmatrix}$$

- We multiply each element by the scalar.

Combination of Operands

$$3 \cdot \begin{bmatrix} 1 \\ 4 \\ 2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} - \begin{bmatrix} 3 \\ 0 \\ 2 \end{bmatrix} /3 = \begin{bmatrix} 2 \\ 12 \\ \frac{31}{3} \end{bmatrix}$$

Matrix-vector Multiplication

$$\begin{bmatrix} 1 & 3 \\ 4 & 0 \\ 2 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 5 \end{bmatrix} = \begin{bmatrix} 1 \cdot 1 + 3 \cdot 5 \\ 4 \cdot 1 + 0 \cdot 5 \\ 2 \cdot 1 + 1 \cdot 5 \end{bmatrix} = \begin{bmatrix} 16 \\ 4 \\ 7 \end{bmatrix}$$

- $m \times n$ matrix, multiplied by a $n \times 1$ matrix (or n-dimensional vector), results in an m-dimensional vector.
- Given the multiplications of a matrix $A_{m \times n}$ by a vector x_n, resulting in a vector y_m, we get each element y_i by multiplying the i-th row of A by x (element-wise) and adding them up.

Hypothesis: Matrix Form

$$h_\theta(x) = \begin{bmatrix} 1 & x_0 \\ 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ \vdots & \vdots \end{bmatrix} \cdot \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}$$

Matrix Multiplication

- Example:

$$\begin{bmatrix} 1 & 3 & 2 \\ 4 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 5 \end{bmatrix} & \begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix} \end{bmatrix}$$

- Part 1:

$$\begin{bmatrix} 1 & 3 & 2 \\ 4 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 5 \end{bmatrix} = \begin{bmatrix} 11 \\ 9 \end{bmatrix}$$

- Part 2:

$$\begin{bmatrix} 1 & 3 & 2 \\ 4 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 10 \\ 14 \end{bmatrix}$$

- Result:

$$\begin{bmatrix} 1 & 3 & 2 \\ 4 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 3 \\ 0 & 1 \\ 5 & 2 \end{bmatrix} = \begin{bmatrix} 11 & 10 \\ 9 & 14 \end{bmatrix}$$

- The number of columns of the left matrix must be equal to the number of rows of the right matrix!
- The multiplication of a matrix $A_{m \times n}$ by a matrix $B_{n \times p}$ results in $C_{m \times p}$.
- The i-th column of the matrix C is obtained by multiplying A with the i-th column of B, for $i = 1, 2, \ldots, p$.

Properties

- **Not** commutative! : $A \times B \neq B \times A$.
- Associative: $(A \times B) \times C = A \times (B \times C)$.
- Identity matrix: $A \cdot I = I \cdot A = A$:
  - $A_{m \times n} \times I_n = A_{m \times n}$.
  - $I_m \times A_{m \times n} = A_{m \times n}$.

$$I_n = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

Inverse and transpose

Inverse

If A is an $m \times m$ matrix, and **if it has an inverse**, then:

$$A \cdot A^{-1} = A^{-1} \cdot A = I_m.$$

- Not all matrices have an inverse.
- *Only* square matrices have inverses.

```
A = [3 4; 2 16]
inv_A = pinv(A) # inverse of A: [0.4 -0.1; -0.05 0.075]
A * inv_A # identity (eye)
```

Transpose

Let A be an $m \times n$ matrix, and let $B = A^T$.

Then B is an $n \times m$ matrix, and $B_{ij} = A_{ji}$.