Week 01

I. Intro

1. Defintions:

- Arthur Samuel, 59' => ML gives computer ability to learn without being explicitly programmed;
- Tom Mitchell, 98': a computer is said to learn from experience (E) with respect to task (T) and some performances (P), if its performances on T as measured on by (P) improves (E).
- Example:
  - Email program watches you mark or no as Spam, and learns how to better filter Spam.
    - T = Classify emails
    - E = Watching you label emails as spam or not spam
    - P = The number of emails correctly classified
  - Playing checkers.
    - T = playing
    - E = play many games
    - P = Probablilty that program will win the next game
- ML algo.: Supervised and unsupervised learning + (bonus: reinforcement & recommender)

2. Supervised ML

Right answers are given and task of algorithm is to produce more right answers.

There's a relationship between the input and the output.

- Example:
- Beast cancer:
  - It is called classification problem: label sizes to 1 for malignant or 0 for benign.
  - pTo Predict results in a discrete output.
  - → classification is about predicting a label.
- Housing price prediction:
  - It is called regression problem to predict continuous valued output (prices).
  - Price as a function of size is a continuous output.
  - Map input variables to some continuous function.
  - → regression is about predicting a quantity.
  - Can turn to into classification problem: output whether the house "sells for more or less than the asking price."
  - Here we are classifying the houses based on price into two discrete categories.
- Age
  - Given a picture, predict the age.

3. Unsupervised ML

- Approach problems with little or no idea what our results should look like.
- Can derive results structure by clustering it based on relationships among the variables in the input.
- No feedback based on the prediction results.
- Example
  - Cocktail party problem: identify human voices and music => Non-clustering.
  - Discover market segments and group customers into different market segments.
  - Group articles into sets about the same stories.
  - Create groups from a collection of 1,000,000 different genes based on lifespan, location, roles, etc.

II. Model and cost function

1. Model representation:

- $x^{(i)}$ = "input" variables = input features.
- $y^{(i)}$ = "output" variables = target variable. => example: predict price (y) from living area (x)
- couple $(x^{(i)}, y^{(i)})$ is called training example.
- couples $(x^{(i)}, y^{(i)}); i = 1..m$ is called training set.
- X space of input values; Y space of output values; $X = Y = \mathbb{R}$
- Supervised learning problem:

- given a training set, to learn $h : X \to Y$ so h(x) ≈ y
- h is called **hypothesis**

[[An algorithm will learn from a training set how to predict y when x is provided using h]]

Such as in our housing example, we call the learning problem a regression problem.

- Goal is to found h and its parameters as $h(x) = ax + b$

2. Cost function

- Cost function = average difference of all results of all x's & y's
  - $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=0..m} (h_\theta(x^{(i)}) - y^{(i)})^2;$
  - m is the number of training examples
- Hypothesis $h_\theta(x) = \theta_0 + \theta_1 x$
  - Parameters: θ_0, θ_1
  - Choose these parameters so $h_\theta(x)$ is close to y for the training couple (x, y)
- Goal: $minimize J(\theta_0, \theta_1)$

Intuition 1:

- Ideally, the line should pass through all the points of our training data set to minimize $J(\theta_0, \theta_1)$.
- $h_\theta(x)$ is plotted : a linear function passing through some points of the training set.
- Plot $h_\theta(x)$ and $J(\theta_0, \theta_1)$

> $\theta_1 = 0$ => $J(0) = 2.3$
>
> $\theta_1 = 0.5$ => $J(0.5) = 0.58$
>
> $\theta_1 = 1$ => $J(1) = 0$
>
> $\theta_1 = 1.5$ => $J(1.5) = 0.58$
>
> $\theta_1 = 2$ => $J(0) = 2.3$

- → Thus as a goal, try to minimize the cost function. In this case, $\theta_1 = 1$ is the global minimum.

III. Parameter learning

1. Gradient descent

- Have some function $J(\theta_0, \theta_1)$
- Want to $minimize J(\theta_0, \theta_1)$
- **Outline**

  - Start with some θ_0, θ_1.
  - Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ to find up a minimum.
- Estimate the parameters in the hypothesis function → Gradient descent

Algorithm:

> repeat until convergence $\theta_j := \theta_j - \alpha \frac{d}{d\theta_j} J(\theta_0, \theta_1)$
>
> for $j = 0$ and $j = 1$.
>
> where j is the feature index number.
>
> hint: calculate values for 0, then for 1, at the end assign values to overwrite $\theta' s$.

In these examples, J is based on the parameter θ_1 → The formula for a single parameter is : $\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$

> Plotting the graphs of $J(\theta_1)$ show that when the slope is:
>
> negative, the value of θ_1 increases: $\frac{d}{d\theta_1} J(\theta_1) \leq 0$
>
> positive, the value of θ_1 decreases: $\frac{d}{d\theta_1} J(\theta_1) \geq 0$
>
> α is positive.

Adjust α to ensure the convergence in a reasonable time.

> if α is too small, gradient descent can be slow to go to the minimum.
>
> if α is too large, gradient descent can overshoot the minimum, or diverge.

- How does gradient descent converge with a fixed step size α?
  - The intuition behind the convergence is that $\frac{d}{d\theta_1} J(\theta_1)$ approaches 0 as we approach the bottom of our convex function. At the minimum, the derivative will always be 0 and thus we get: $\theta_1 := \theta_1 - \alpha * 0$
- Facts:
  - To make gradient descent converge, we must slowly decrease α over time.
  - Gradient descent is guaranteed to find the global minimum for any function $J(\theta_0, \theta_1)$.
  - Gradient descent can converge even if \alphaα is kept fixed. (But \alphaα cannot be too large, or else it may fail to converge.)
  - For the specific choice of cost function $J(\theta_0, \theta_1)$ used in linear regression, there are no local optima (other than the global optimum).

Gradient descent for linear regression:

- replace J by its value in the algorithm:

Algorithm:

> repeat until convergence
>
> $\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{1..m} (h_\theta(x_i), y_i)$

> $\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{1..m} ((h_\theta(x_i), y_i) x_i)$
>
> m is the size of training data

Batch gradient descent:

- Each step of gradient descent uses all the training examples.

IV. Linear Algebra

- Matrices are 2-dimensional arrays
  - A vector is a matrix with one column and many rows
  - 1-indexed or 0-indexed
  - Matrices are usually denoted by uppercase names while vectors are lowercase.
  - “Scalar” means that an object is a single value, not a vector or matrix.
  - Matrices are not commutative: A∗B ≠ B∗A
  - Matrices are associative: (A∗B)∗C = A∗(B∗C)