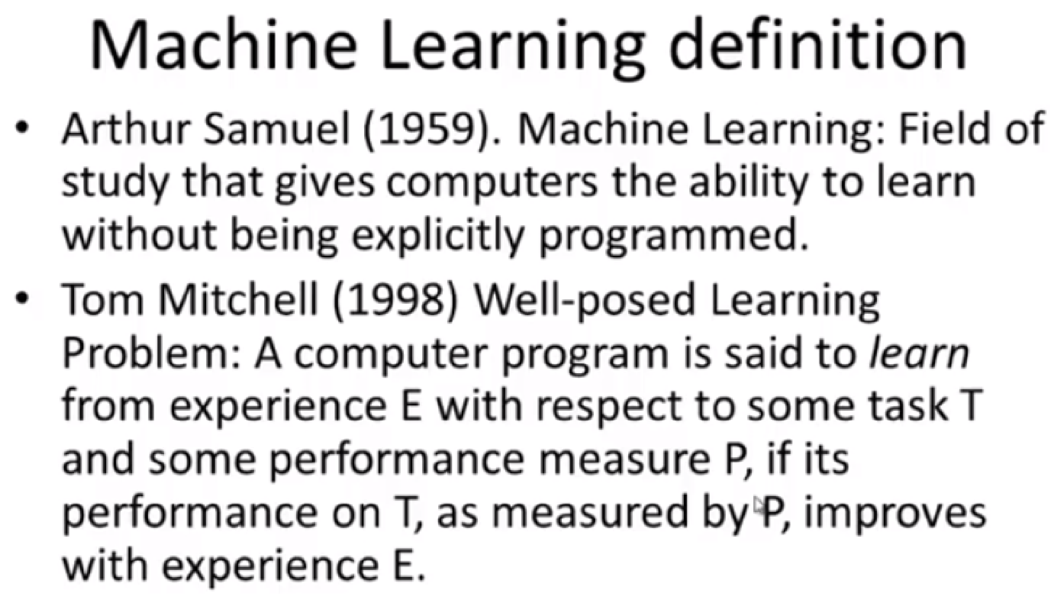
machine learning_week1

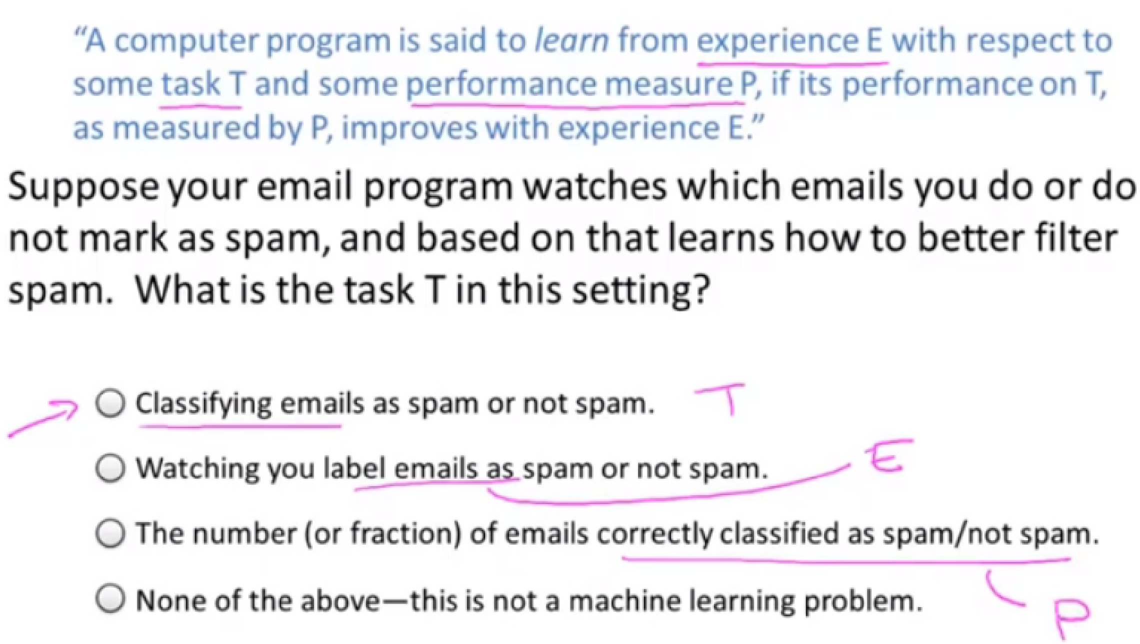
Notebook:	artificial intelligence		
Created:	18/09/24 09:42	**Updated:**	18/11/04 06:42
Author:	2440027575@qq.com		

1.introduction:

- glance:
  - learn algorithms and theroy,
  - make it applied!
    1. Data mining
    2. auto programming
    3. personal customization
  - future:understand human brain
- ML def:



  -
  - **focus on:experience_E/task_T/performance_P**
    - to understand the second defination: take question as example



    -
- algorithm: supervised learning/unsupervised learning

1.3 supervised learning

- how to fit data set? choose the right model,
- supervised learning： data set+labels
  - regression: predict results within a **continuous** output make continuous function(map) to fit the data set
  - classification： predict results in a **discrete** output **map** input variables into discrete categories
- more features to make model accurate.

1.4unsupervised learning

- dataset without labels→ derive structure: **clustering algorithm/Non-clustering**
  - example： google news
  - more applications: genomics / social network analysis / market segmentation / astronomical data analysis
- e.g. Cocktail party problem algorithm
  - learn more: cocktail party.(simplify:brain's ability to focus attention while in chaos.)

---

week.1_Linear regression with one variable

Model and Cost Function

project: house price prediction.

- before practice： model representation
  - $(x^{(i)}, y^{(i)})$:training example. The superscript is begin **with 1 not 0**
  - X: the space of input values.
  - hypothesis: the function we used in supervised learning (called this for historical reasons)
- **Cost Function**
  - used to: measure the accuracy of hypothesis function.
  - here use **"Squared error function"**

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

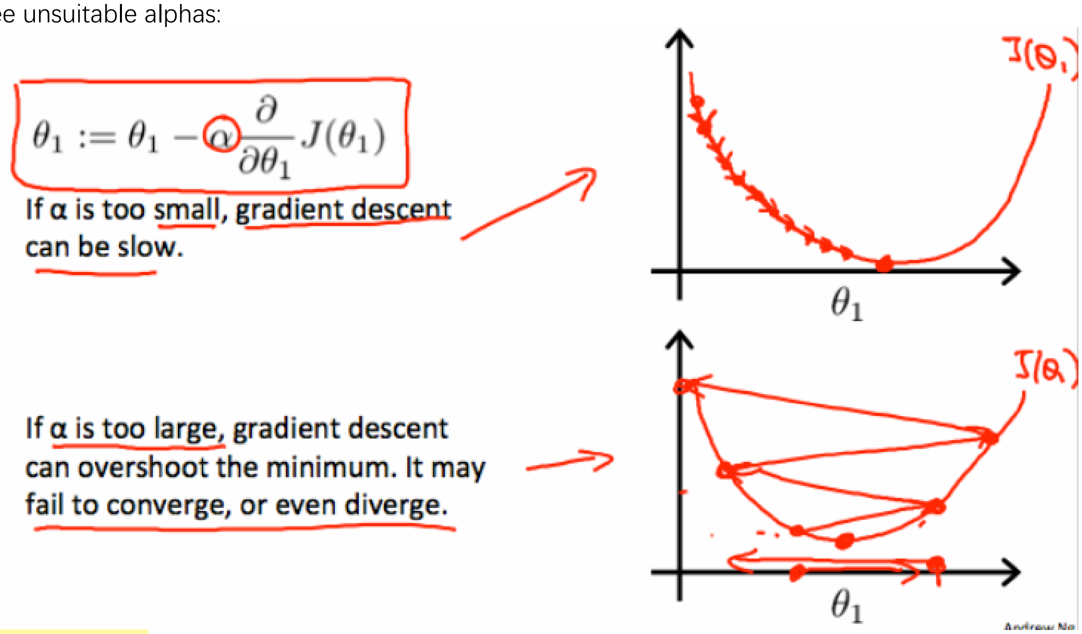
  - **goal: minimize J(θzero,θone)**
    - θzero、θone equal to the parameters in y=b+kx
    - *why "1/2m" ? That is a parameter we choose to amplify our result
  - Intuition I: let θzero=0, make cost function simpler.
  - Intuition II: visualize the hypothesis with contour plot or 3-d surface. **remember: visualization** just help us to better understand how cost function work. But our aim is **automatically algorithm to find the value of thetas.**

Parameter Learning

- Gradient Descent
  - first intuition: find the fast way to down a hill. We need to find the **sloping** road.
    - if we have different initialization, we may reach different destination. —— just reach local optimum.
    - (reflection: sounds a bit like Greedy algorithm)
  - **algorithm:**

$$\text{repeat until convergence } \{$$
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad \text{(simultaneously update } j = 0 \text{ and } j = 1)$$
$$\}$$

-
- converge to a **local minimum**.($\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$ =0,so theta j would not change)
- ":=" means assignment. Just equally means "=" in programming
- α is a parameter to control the rate of convergence
  - a suitable alpha is important for convergence.
  - see unsuitable alphas:



- **Simultaneous update** is important. We update all the thetas in one loop.(caused that if we make a serial update for theta, the theta(j+1) will use the new theta(j) instead of the original one)
- back to our "squared error function":
  - by calculate the partial derivative of theta j,we got

$$\text{repeat until convergence: } \{$$
$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)$$
$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} ((h_\theta(x_i) - y_i) x_i)$$
$$\}$$

  - details:
    - this is a convex quadratic function,means that it only have a **global optimum**, so we don't need to worry different optimums.
    - learn term: **batch gradient descent**
      - method looks at every example in the entire training set on every step

What's more:

- the gradient represents the **slope** of the tangent of the graph of the function.

- gradient vector: points in the **direction** of the **greatest rate** of increase of the function,☜help us to find the fast and accurate direction to convergence.

sum up:

1. observe the data set, find out that we can use linear function to fit it.
2. to fit better☞use Cost function. This case we choose **Squad Error Function.**
3. to get suitable theta 1&2☞algorithm: **gradient descent,** repeat to update theta 1&2 until convergence.(why this: gradient points the fastest direction to increase the function.

---

linear algebra review

- matrix and vector
- addition and scalar multiplication.
- matrix multiplication
  - in ML, use matrix to compute is much more efficient than iteration.
  - example.