1. 人工智能算法简介

1.学习目标

1 ．了解人工智能能做什么事？

2 ．学习准备知识: Python编程基础，**线性代数**，概率基础知识．

2. 人工智能鸟瞰

人工智能(Artificial Intellegence, AI)可以做什么？ 首先，我们要给大家简要介绍人工智能的能做什么事情．人工智能的范围非常广泛,从人工智能的历史，搜索算法的建立，设计游戏, 解决游戏难题，到限制条件问题都值得学习. 机器学习算法是人工智能里的核心．人工智能可广泛应用在自然语言处理，机器人学，机器视觉，语音分析，量化交易等等领域. 用Python语言编程来解决人工智能问题是一个值得学习的技术．下面分别介绍一下各种常见算法．

2.1搜索

最基本的算法就是搜索．有许多中搜索方法可以使用比如盲目搜索（暴力搜索，无脑搜索，uninformed search)， 提示性搜索(又叫**启发性搜索**)，对抗搜索(游戏)等．第二类话题就是马科夫决策过程和强化学习. 它们有一系列的应用，如自然语言处理，机器人，机器视觉等. 现在我们一一讨论人工智能里的各个话题．

先来看理性智能体. 我们研究人工智能的目的是设计智能体，它们可以**感知**其环境并且**作用**到环境上，从而实现其目标或者任务．一个智能体可以视为一个函数F(x), 该函数从感知到的环境映射到一个作用在环境上的动作. 理性智能体，就是做正确的事情的智能体．何为正确的事呢？就是智能体的表现达到最优，即所谓**性能度量**(performance measure)最大化．人工智能(AI)在给定的计算条件下，使得性能度量达到最大化．这就是AI的目的．要使得性能度量最大，可以从硬件和软件两方面优化改进，我们这里只讨论软件方面．

搜索智能体可以帮助我们从已知点出发找到目标点．典型的例子是走迷宫，从某个给定起点和终点，找出一条路线使得我们能从起点到达终点. 智能体会思考为了达到目的该如何做．

智能体要做的就是定义出到达目标点的动作或动作序列（路径）.一条路径会有不同的代价和深度（此处指的是通过该路径找到的解在搜索树中的深度）. 最常见搜索方法可分为有两大类. 盲目搜索并不用某领域的知识，它包括的技术有广度优先搜索，深度优先搜索，均匀代价搜索等． 启发式搜索运用了一些如何更快地到达目标的经验法则或启发式信息，这类搜索法包括**贪婪搜索法**， A*搜索法, 等等. 搜索算法的例子包括八皇后问题．八皇后问题是指，我们在64个格子的国际象棋棋盘上适当地放置 8 个皇后，使得它们横向，纵向，对角向都不＂共线＂．这就是要从约百万亿种可能的状态中，搜索出满足以上约束条件的状态来．另一个典型的搜索算法的例子就是路线搜索． 给定包含一些城市的地图，地图可以用图结构来表示：城市用结点表示，城市之间的可能的路线用线表示．

两座城市之间的距离标记在它们的连线上．假设你想要从北京到马尔代夫．要达到此目的，你有不同的路线可走．搜索智能体的目的就是为了探索这些可能性，并找出最好的路线．不同的路线需要不同的花费，花费可以用这些连线的长度(比如，单位为千米的数)　这里，搜索智能体的角色就是找到从北京到马尔代夫的路线，同时找出最好的路线．

对抗搜索

AI的另一个重要话题是对抗搜索，或称游戏．实际上，已经有一些游戏可以用AI解决.我们看到国际象棋，象棋以及智力问答竞赛等等.　基本思想就是设计智能体来玩游戏，并且与对手对抗.　这里面的重要概念包括**mini max 算法**, **alpha beta pruning** 和**随机游戏**.　这里略过．在下棋这一领域，计算机(或者Chinook)能打败任何人类. 对抗搜索的另一个名字就是游戏(games). 对抗(adversarial)意味着存在一个我们无法控制的对手.

对比: 游戏和搜索

游戏也是一个搜索问题，但是最优解不是一个引向目标的动作序列．而是一个帮助我们赢得比赛的策略. 也就是说，如果对手做A操作，那么我们就做甲操作，对手做B操作，我们就相应地施行乙操作,如果对手做C操作，我们就做丙，等等. 这种策略可以写成规则，好比下棋时候的口诀，但是它非常地枯燥和冗长．

2.2 机器学习

机器学习是当今AI的必然趋势．根据机器学习的杰出专家Tom Riccio所说，机器学习是关于如何创建能从数据和观测中提高我们的经验的计算机程序.　你想要能教计算机如何学习并如何提高经验．这是机器学习之核心．

机器学习主要可分为监督学习，非监督学习和强化学习等．

第一类学习，是**监督学习**．当你有标签时，那么你在做的就是监督学习．这些标签可以是任意连续的数值，也可以是离散的值（如文本）．如果标签只能取两个值，那么我们就在讨论二进制分类．我们想要建立的是一个函数，给定一个输入集合或是实例的描述，得出输出集合.　例如，假设我们有银行顾客的信息：他们的年龄，性别，职位，工资等等. 对每个顾客，我们有一个标签：是否有信用卡．又如，电子邮件是否是垃圾邮件也属于标签的例子．这样例子还有很多.　因此，我们有所谓的正例子和负例子．它们可以分别用1, 0 表示. 如何找出分开这两类特征的边界往往是这类问题的目标．这是监督学习的典型例子．因为已经告诉了我们谁是正例子，谁是负例子了，所以称之为监督学习．监督学习包括分类，k近邻预测，神经网络，线性回归，boosting等等方法．

第二类学习，就是**非监督学习**(unsupervised). 非监督学习中，我们不必为实例做标签. 也就是说，我们的数据集中的实例是没有标签的．例如，我们有人口或顾客的数据，却没有任何类型的标签与之相联系．因此，这里的要解决的问题就是：给定这些数据点，我们能否找出这些实例的数据点的聚类(clusters)吗? 我们要寻找一个函数F用以把输入集合X映射到聚类的集合．这完全是非监督的算法．关于非监督学习，我们有很多不同的方法来实现．最主要的为K均值聚类, 高斯混合, hierarchical agglomerative clustering, spectral clustering, DBSca等．

在**强化学习**中，我们设计智能体在随机或特定环境中演化．智能体从强化或延迟奖励中学习．它是除监督学习和非监督学习之外的又一类学习．它是一类用于在输出结果为随机值的决策问题中的学习方法,它包含了一个能连续计划，学习并影响其环境的智能体．强化学习的驱动力是最大化奖励． 在对抗，博弈，游戏，投资，交易等等领域都可以应用．

2.3 约束问题

约束问题，其实是搜索问题，它不关注搜索的路径，而只关心**目标**，例如解一个游戏. 一般地，问题用**变量**来表示，而不是用态来表示．约束问题(constraint satisfaction problem)的典型例子是数独游戏(Sudoku)．例如，在一个9行9列的格子盘中，横向一排格子我们称为一行，纵向的一排格子称为一列．要达到的目标是各行的数字都只能取从 1 到9的数字，各列亦如此，且在 9 个 3 X 3 的小九格的数字也只能是从1到9的不重复的数字．你可以表示这个问题为一个CSP问题．每一个格子可视为一个变量．加在这些变量上的约束条件就是所有3X3格点内，所有行，所有列中的数字都只能是从1到9的不重复的数字．问题的解就是去寻找为变量赋值的方法，以满足这些约束条件．

2.4 逻辑智能体

人工智能的又一个重要领域是逻辑智能体．在**逻辑智能体**中，逻辑被用来构建我们所在的世界的模型． 很早以前，布尔发明了命题逻辑(propositional logic), 并且，一阶逻辑有一个固定的语法．通过句子的逻辑表达式，我们用符号和连接词(connectives)来为世界建立模型. 关于逻辑智能体的实例，读者可参考Prolog相关文章．

2.5 自然语言

人工智能可以用于自然语言处理，在这个领域我们关注计算机和人类语言之间的相互作用，我们关注视觉，感知于图像处理，并致力于建立AI机器视觉．这里的任务就是从任务中提取信息．这样的任务包括操纵，谈判，模式识别等．

2.6 操纵物理世界

最后，机器人学关注如何让人工智能体操纵物理世界．比如，实现自动驾驶，识别人脸，机器人的自我运动等．

如果你想学习人工智能算法，那么你的准备知识应该包括一些编程知识，线性代数和对概率的理解．今天，我们先介绍线性代数基本知识．

3.线性代数基本概念

3.1 矢量

数学上看，矢量空间的元素，或矢量空间中的点，就是**矢量**．更具体地看，有序的值的序列就构成了一个矢量．一般用小写黑体字母表示，如

$$\mathbf{x} = \begin{pmatrix} 2 \\ 3 \end{pmatrix} = (2,3)^T \tag{36}$$

是二维空间中的矢量．我们默认都采用笛卡尔坐标系．又如

$$\mathbf{y} = \begin{pmatrix} 1.0 \\ 1.0 \\ 1.0 \end{pmatrix} \tag{1}$$

是三维空间中的矢量．矢量的元素用下标标识，

$$x_1 = 2, x_2 = 3. \tag{2}$$

注意:

1. 举例: 我们所在的三维空间可以近似看作是一个三维矢量空间．取定某点为原点，那么空间中每一点就是该空间中的一个矢量．所有的矢量就构成了这个三维空间本身．

2. 在学习Python时，区分列表[2,3], 元组(2,3)和矢量

$$\mathbf{x} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}.$$

3.2 矢量的基本操作

3.2.1 矢量的加法：

$$\mathbf{x} + \mathbf{z} = \begin{pmatrix} 2 \\ 3 \end{pmatrix} + \begin{pmatrix} 1 \\ -2 \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \end{pmatrix} \tag{3}$$

一般地，对于两个二维矢量求和，结果为对应坐标相加

$$\mathbf{a} + \mathbf{b} = \begin{pmatrix} a_1 + b_1 \\ a_2 + b_2 \end{pmatrix} \tag{4}$$

注意：

1. 只有维度相同的两个矢量才可以相加；
2. 对于高维空间中的矢量的求和，可以类推．

3.2.2 矢量的数乘

已知常数\lambda,和n维矢量{\bf x}．数乘定义为

$$\lambda \mathbf{x} = \lambda \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} \lambda x_1 \\ \lambda x_2 \\ \vdots \\ \lambda x_n \end{pmatrix}. \tag{5}$$

例如，有矢量

$$\mathbf{x}=\begin{pmatrix}1\\0\\0\end{pmatrix},$$

数乘为

$$3\mathbf{x}=3\begin{pmatrix}1\\0\\0\end{pmatrix}=\begin{pmatrix}3\\0\\0\end{pmatrix};$$

矢量为

$$\mathbf{y}=\begin{pmatrix}1\\1\end{pmatrix},$$

$$0.707\mathbf{x}=0.707\begin{pmatrix}1\\1\end{pmatrix}=\begin{pmatrix}0.707\\0.707\end{pmatrix};$$

说明：

1. 当常数为绝对值大于 1 的数时，数乘将矢量**伸长**，当常数为绝对值小于 1 的数时，数乘将矢量缩短．
2. 常数可以是**实数**，也可以是复数．(本课程中，大家可以不要考虑常数为复数的情形)

3.2.3 矢量的长度

矢量的长度，又称模. 现在举例说明之．例如,对于

$$\mathbf{x}=\begin{pmatrix}3\\4\end{pmatrix},$$

$$|\mathbf{x}|=\sqrt{3^2+4^2}=5$$

通常，**矢量长度**取为各元素的平方和之平方根．

3.2.4 矢量的点积

两个矢量的点积，又叫标量积，定义为两个矢量的对应元素的乘积. 例如，对于二维矢量x和y,其点积

$$\mathbf{x}\cdot\mathbf{y}=x_1y_1+x_2y_2. \tag{6}$$

举例，

$$x=\begin{pmatrix}4\\5\end{pmatrix};y=\begin{pmatrix}1\\0\end{pmatrix};\mathbf{x}\cdot\mathbf{y}=4\times1+5\times0=4 \tag{7}$$

注意：

1. 矢量的点积的结果为一个数，而不是矢量．
2. 矢量的点积满足**交换律**：

$$\mathbf{x} \cdot \mathbf{y} = \mathbf{y} \cdot \mathbf{x}. \tag{8}$$

3. 在**书写**矢量点积的表达式中，点积符号(圆点)不可省略:

$$\mathbf{x} \cdot \mathbf{y} \neq \mathbf{xy} \tag{9}$$

4. 举例：一个人有年龄，性别，身高，体重，职业，收入，婚否，母语，国籍，宗教信仰．这些特征中的每一个都可以看成是一个维度．每个维度上的信息，可以用数字来编码．最后，对于这个人我们可以得到十维空间中的一个矢量！（注意：严格来说这些量具有不同的单位，这样组成的量不是矢量．但如果我们选择适当的单位，合适地编码，我们可以把它们看成矢量.）

$$V_{\text{person}} = \begin{pmatrix} 24 \\ F \\ 170 \\ 60 \\ OL \\ 5.5 \\ N \\ Chinese \\ CN \\ Tao \end{pmatrix} = \begin{pmatrix} 24 \\ 0 \\ 170 \\ 60 \\ 3 \\ 5.5 \\ 0 \\ 1 \\ 3 \\ 0 \end{pmatrix} \tag{10}$$

4. 同理，对于网页上的一篇文章，交易数据，甚至任何一个事件，我们都可以将它表示成一个矢量．例如，利用bag-of_words技术，我们可以把两篇文章分别表示成矢量，通过计算这两个矢量之差，或者两个矢量的点积，我们就可以判断两篇文章的相似度，从而可以有助于我们进行文章的归类，鉴别剽窃与否等等．

习题1：(30分)

3.3 矩阵

由特定行数和列数的数构成的数学对象，称之为矩阵．形如

$$X = \begin{pmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \end{pmatrix} \tag{11}$$

者，就是一个2×3矩阵. .

例如，像

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{12}$$

由2行2列的数排列在一起构成的数学对象，就称为**2×2矩阵**．　一般地，一个M×N的矩阵就是M×N个数排成M行N列的一个数学对象. 这里M,N都是大于或等于 1 的整数．　一般，矩阵用大写字母表示．在此，X, Z都是2行 2 列的矩阵．举例来说，一个矩阵描述的是对一个对象的操作．这里所说的对象，往往可以用一个矢量来描述．　矩阵作用在矢量上，会得到一个数，一个新矩阵，或者**另一个矢量**．矩阵本身也可以描述一个对象 / 状态，为之建模．一个M维矢量可看是一个M×1矩阵．

矩阵的分量用两个下标来标注，其中，第一个下标为行数，第二个下标表示分量所在的列数．在程序中，矩阵A的第i行第j列的分量，往往写作

$$A[i][j] \tag{13}$$

3.4 矩阵的基本操作

3.4.1 矩阵的和

只需记住一点：若已经矩阵A和B，它们具有相同的形状(size)， 则A与B的和由下面式子定义

$$(A+B)_{ij} = A_{ij} + B_{ij} \tag{14}$$

分量得出，则得到A+B.

3.4.2 矩阵的数乘

矩阵的数乘，也叫矩阵的标量积．计算公式为：

$$(\lambda A)_{ij} = \lambda A_{ij} \tag{15}$$

例如

$$5\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 5\times 1 & 5\times 0 \\ 5\times 0 & 5\times(-1) \end{pmatrix} = \begin{pmatrix} 5 & 0 \\ 0 & -5 \end{pmatrix} \tag{16}$$

3.4.3 矩阵的乘法

不是任意两个矩阵都可以定义矩阵乘法．只有当第一个矩阵的列数等于第二个矩阵的行数时才能定义．

举一例说明. 例如

$$AB = \begin{pmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{pmatrix} = \begin{pmatrix} 0\times 0+1\times 2+2\times 4 & 0\times 1+1\times 3+2\times 5 \\ 3\times 0+4\times 2+5\times 4 & 3\times 1+4\times 3+5\times 5 \end{pmatrix} = \begin{pmatrix} 9 & 13 \\ 17 & 40 \end{pmatrix}$$

2 × 3 矩阵乘 3 × 2 矩阵得到一个 2 × 2 矩阵．

再举个例：

$$X\mathbf{a} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0\times 1+1\times 0 \\ 1\times 1+0\times 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{17}$$

2 × 2 矩阵乘 2 × 1 矩阵得到一个 2 × 1 矩阵，即结果为一个新的矢量．

再举两个例：

$$Z\mathbf{a} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1\times 1+0\times 0 \\ 0\times 1+(-1)\times 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{18}$$

$$Z\mathbf{b} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1\times 1+0\times 1 \\ 0\times 1+(-1)\times 1 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \tag{19}$$

一个 2 × 2 矩阵(Z)乘 2 × 1 矩阵**a**得到一个 2 × 1 矩阵，即结果为*原来*的**矢量**！从(21)式可以看出，当Z与**b**相乘时得到的结果是一个*新的*矢量.

矩阵乘法的性质：

1. 对于一般矩阵，矩阵的乘法**不满足交换律**．对于矩阵A, B,

$$AB \neq BA \tag{20}$$

   举例子．假设你在赤道上一点**x**．　你先向东走1000km,再向北走1000km到达**y**；　若从该点先向北走1000km后，再先向东走1000km，到达**z**.　最终你会发现：**y，z**两点并不重合----两种走法到达了不同的地方！　又如，两次转动魔方的顺序交换一下，得到的魔方的形态是可能不一样的.

2. 矩阵乘法**满足结合律**．对于矩阵A,B,C,

$$(AB)C = A(BC) \tag{21}$$

3.4.4 转置矩阵

矩阵A的**转置矩阵**定义为

$$(A^T)_{ij} = A_{ji} \tag{22}$$

例如

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}; \Rightarrow A^T = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} \tag{23}$$

二维矢量是 2 × 1 矩阵，因此也可以计算其转置矩阵．例如

$$\mathbf{x} = \begin{pmatrix} 4 \\ 5 \end{pmatrix}; \Rightarrow \mathbf{x}^T = \begin{pmatrix} 4 & 5 \end{pmatrix} \tag{24}$$

应用：利用矩阵的**转置矩阵**，矢量的点积可以矩阵相乘形式来表示．

$$\mathbf{x} \cdot \mathbf{y} = \begin{pmatrix} 4 & 5 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 4 \times 1 + 5 \times 0 = 4 \tag{25}$$

3.4.5 逆矩阵

单位矩阵I的定义：元素满足条件

$$I_{ij} = \begin{cases} 1 & \text{for } i = j \\ 0 & \text{for } i \neq j \end{cases} \tag{26}$$

的N×N矩阵(方阵)．单位矩阵的性质就是：任何矩阵乘以I都仍然得该矩阵本身:

$$AI = IA = A \tag{27}$$

例如：

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \tag{28}$$

对于一个方阵A，它的逆矩阵A^{-1}定义为

$$A^{-1}A = I \tag{29}$$

对于奇异矩阵，其逆矩阵不存在．

3.4.5 解线性方程

矩阵可用来解决线性方程组． 什么是线性方程组呢？

看一个例子．鸡兔同笼是中国的数学名题之一. 参考1500年前《孙子算经》：今有雉兔同笼，上有三十五头，下有九十四足，问雉兔各几何？

解法1:

设有鸡兔分别有x,y只，可列方程组：

$$x + y = 35$$
$$2x + 4y = 94$$

你可以用代入消元法解这个方程组．

解法2:

请小鸡和小兔都举起自己的两只脚．那么它们将有70只脚离开地面．于是，地面将只剩下94-70=24只脚．现在地上已经没有鸡脚了．所以只有小兔的脚，所以小兔的数目为24/2 = 12.　故小鸡的数目为35-12 = 23.

解法3:

我们也可以把这个方程组写成矩阵形式：

$$\begin{pmatrix} 1 & 1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 35 \\ 94 \end{pmatrix} \tag{30}$$

也就是这样的形式：

$$A\mathbf{x} = \mathbf{b} \tag{31}$$

我们只需要求出矩阵A的逆矩阵，然后把它在上式两边同时乘上就行了！

要想解出x, 可以在上式两边从左边乘上A矩阵的逆矩阵：

$$A^{-1}A\mathbf{x} = A^{-1}\mathbf{b} \tag{32}$$

也就是

$$\mathbf{x} = A^{-1}\mathbf{b}$$

这是很**通用**的解法！

又如，线性方程组

$$\begin{aligned} 2x + 2y - z &= 4 \\ -x + 2y - z &= -6 \\ -3x + y + 2z &= -3 \end{aligned}$$

可以写成如下形式

$$\begin{pmatrix} 2 & 2 & -1 \\ -1 & 2 & -1 \\ -3 & 1 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 4 \\ -6 \\ -3 \end{pmatrix} \tag{33}$$

即

$$A\mathbf{x} = \mathbf{b} \tag{34}$$

要想解出x, 可以在上式两边从左边乘上A矩阵的逆矩阵：

$$A^{-1}A\mathbf{x} = A^{-1}\mathbf{b} \tag{35}$$

也就是

$$\mathbf{x} = A^{-1}\mathbf{b}$$

多元一次方程组当然也可以这样来求解！现在，我们就找到了一种求解线性方程组的一般方法. 我们可以把求解线性方程组问题用矩阵的语言重新表示出来. **解线性方程组的关键步骤就是: 求其系数组成的矩阵的逆矩阵**！

4. 总结

人工智能(AI)是一个宏大的学科，因为它包含很多方面．包括其组成的复杂性，语言和划分等.　AI也是一个广阔的学科，对人类自身及社会有极其重大的影响．

AI系统不一定要模拟人类或自然之机制．例如，如鸟一样，飞机是受到鸟的飞行的启发．飞机可以飞行，但机制完全不一样．又如，人类如何思考也远在我们当前科学能理解的范围之外．大家记住，我们AI课程的核心方法就是：**能做正确的事情的智能体**，而不是一味地模拟人类或自然之机制.

尽管完全的理性是不能做到的，因为环境太复杂．我们甚至不知道什么是**正确的事情**．但我们要尽量做到正确，这里，确定性丧失了，复杂因素变多了，**概率和统计**的作用非常关键，因而我们有必要了解有关概率和统计的基本知识．(几天以后课程再介绍)

在媒体和公众那里经常被问起的一个问题就是：是否AI是人类的威胁？　的确，AI可能让我们产生一些对未来不确定的感觉，并且人们会想是否能保住自己的工作，AI系统是否会接管地球导致人类灭绝．(包括Stephen Hawkins，他告诉BBC，发展完全AI导致人类的灭绝. 他对人工智能思考得比较多，因为他亲身体验过它们在用于交流时是多么强大.)　相比这些问题挑战，我们的社会有更多机遇和进步的可能．我相信人们有信心能解决．未来社会，每一个人都会和AI息息相关．AI是激动人心又繁荣的学科领域，它将改变我们的生活方式．正如计算机和网络改变了我们与父辈祖辈的生活方式一样．人人都可认识AI,利用AI,**为AI做贡献**．

希望大家要将不同的学科的知识融汇贯通，做一个跨学科的有创造力的人．AI领域，就可以让你有机会把自己培养成一个有创造性的跨学科工作者----人才． 举例，时间 - 空间，引力 - 加速度，物质 - 能量, 人员管理系统 / 金融 - 图论，游戏 - 树 / 栈 / 队列 / 堆，设计 / 规划 - 局部搜索，...

5. 习题

习题地址： https://github.com/hg08/ai_lecture/blob/master/week1/ex_week1_1.pdf

0：(**10分**)写出**4维**矢量**a**,**b**的和

$$\mathbf{a}+\mathbf{b}$$

的表达式，用它们的分量表示．

1：(**30分**)两矢量点积可以用两矢量的长度，以及夹角的余弦表示为

$$\mathbf{x}\cdot\mathbf{y}=|\mathbf{x}||\mathbf{y}|\cos\theta.$$

已知

$$\mathbf{a}=\begin{pmatrix}4\\0\end{pmatrix},\mathbf{b}=\begin{pmatrix}-4\\0\end{pmatrix},\mathbf{c}=\begin{pmatrix}4\\3\end{pmatrix}.$$

求矢量**a**与矢量**c**的夹角；求矢量**b**与矢量**c**的夹角．用python代码实现这个功能．程序名取为"**dot_product.py**". 使得运行命令

python dot_product.py

能够正常运行．要求输出结果到名为"**output_**.txt**"的文件中(**为你的姓名，可以用拼音). 文件中包含两行内容，格式如下：

a与c夹角的余弦: ***

a与c的夹角(rad): ***

a与c的夹角(°)： ***

说明：

1. 有模板可用，下载地址: https://github.com/hg08/ai_lecture/blob/master/week1/dot_product.py
2. 模板不是必需的，你可以完全自己写．(以后省略此说明)

2: (**20分**)

(1)计算下面各式,

$$-1\begin{pmatrix}1\\1\\0\end{pmatrix};-2\begin{pmatrix}1\\0\\1\end{pmatrix}$$

(2) 并对计算结果再求出其模．

(3) 由对矢量做数乘后所得的结果,试解释上面的数乘(常数为**-1**, -2)分别表示对原矢量做了什么操作？

3: **(10分**)矢量和矩阵有什么区别? 请列出来．矢量和矩阵在Python里的实现方式有什么共同点？

4: (**30分**) 假设我们有一本词典,　里面只有五个单词 [a, b, c, d, e]. 这里有三个文档：

文档A: [a,b, b, d,d,e]

文档B: [b,b,b,e,e,e,d,a]

文档C: [d,b,b,e]

使用bag-of-words模型将每个文档表示成五维向量,每个维度上的分量分别代表a,b,c,d,e这五个单词在文档中出现的次数. 例如文档A可表示为

$$\mathbf{a}=\begin{pmatrix}1\\2\\0\\2\\1\end{pmatrix}.$$

(1)将文档B和文档C按照上述模型表示成矢量.

(2) 若定义两矢量夹角的余弦值为这两个矢量所对应的文档的**相似性**．计算文档A与文档B的相似性，计算文档A与文档C的相似性.