

<-----

学习目标:

- 1、了解人工智能的概念以及机器学习的概念, 知道其背后运行的方式, 对人工智能在当代的发展有初步的认识。
- 2、通过教案以及检索相关资料, 学会搭建运行 Python 的集成开发环境, 学习基本的 Python 编程方式以及编程的思维。
- 3、学习机器学习算法, 对机器学习进行初步的实践

学习平台: CSDN、Github、Stack Overflow、知乎、Coursera、b 站等

----->

人工智能基础知识

一、人工智能概述

1、什么是人工智能

人工智能 (Artificial Intelligence, 简称AI) 是一门研究如何使计算机模拟和展现出人类智能的学科。它模仿了人类的思维过程, 通过学习、推理、问题解决和自主决策等方式, 使计算机能够执行一系列复杂任务。人工智能是计算机科学的一个分支, 它试图了解智能的实质, 并生产出一种新的能以人类智能相似的方式作出反应的智能机器。

人工智能的发展历程可以追溯到上个世纪50年代。在过去的几十年中, 人工智能已经取得了长足的进步, 包括机器学习、深度学习、自然语言处理、计算机视觉等诸多领域。如今, 人工智能已经深入到我们的生活中, 影响着我们的社会、经济和文化。

人工智能是对人的意识、思维的信息过程的模拟。从模拟人的角度来说, 可以通过认知建模的过程让机器学会像人一样思考, 可以通过图灵测试为目标让机器具有和人一样的行为。总的来说, 人工智能不是人的智能, 但能像人那样的思考。



2、人工智能发展历史

人工智能的发展可以分为以下几个阶段：

(1) 符号主义阶段：在二十世纪五六十年代，人工智能的研究主要集中在符号推理和逻辑推理方面。这个阶段的研究方法主要通过编写规则和知识库，让计算机能够进行基于逻辑规则的推理。

(2) 连接主义阶段：在八十年代初，随着神经网络和连接主义理论的兴起，人工智能的研究开始转向模拟人脑的神经网络。连接主义的研究方法通过构建大规模的神经网络，使计算机能够从数据中学习和泛化。这一阶段的代表性成果是反向传播算法的提出。

(3) 统计机器学习阶段：进入九十年代，统计机器学习成为人工智能研究的主要方向。统计机器学习利用统计模型和概率方法，从大规模数据中学习模式和规律，并用于分类、聚类、回归等任务。这一阶段的代表性算法包括支持向量机、朴素贝叶斯、决策树等。

(4) 深度学习阶段：2010年以后，随着深度学习的兴起，人工智能进入了一个新的发展阶段。深度学习利用深层神经网络模拟人脑的神经结构，通过多层次的网络结构进行特征提取和表示学习。这使得计算机可以更好地理解和处理图像、语音、自然语言等复杂数据。深度学习的代表性算法包括卷积神经网络、循环神经网络和生成对抗网络等。

(5) 当前阶段和未来发展：目前，人工智能正处于一个快速发展的阶段。除了深度学习，人工智能还涉及到自然语言处理、计算机视觉、强化学习等多个领域。人工智能也在各个行业得到广泛应用，包括医疗、金融、交通、教育等。未来，人工智能还将面临更多的挑战和发展机会，如进一步提升模型的效率和鲁棒性，解决数据隐私和安全问题，实现更加智能和可持续的发展。



3、人工智能的分类

(1) 弱人工智能 Artificial Narrow Intelligence(ANI): 弱人工智能是擅长于单个方面的人工智能。

(2) 强人工智能 Artificial General Intelligence(AGI): 人类级别的人工智能。强人工智能是指在各方面都能和人类比肩的人工智能, 人类能干的脑力活它都能干。

(3) 超人工智能 Artificial Super Intelligence(ASI): 知名人工智能思想家 Nick Bostrom 把超级智能定义为“在几乎所有领域都比最聪明的人类大脑都聪明很多, 包括科学创新、通识和社交技能。

二、机器学习概述

1、什么是机器学习

机器学习 (Machine Learning) 是一种人工智能的分支领域, 旨在使计算机能够从数据中学习和提高性能。它通过构建数学模型和算法, 让计算机能够从数据中自动学习规律和模式, 并做出准确的预测和决策。

机器学习的核心思想是基于数据进行模式识别和模型构建。它的工作流程一般包括以下几个步骤:

(1) 数据收集和预处理: 收集和整理相关的数据, 并进行数据清洗和预处理, 以确保数据的质量和准确性。

(2) 特征选择和提取: 选择有相关性的特征并提取出数据中的有用信息, 用于建立模型和进行预测。

(3) 模型选择和训练: 选择适合问题的机器学习算法和模型, 并利用已有的数据进行训练。训练过程中, 模型会根据输入数据进行参数调整和优化, 以最大程度地减少预测错误。

(4) 模型评估和优化: 使用测试集评估模型的性能, 并根据评估结果进行模型优化和调整。这有助于提高模型的泛化能力, 即对未见过的数据进行准确的预测。

(5) 预测和决策: 利用经过训练和优化的模型, 对新的数据进行预测和决策。这可以是分类问题 (将数据分为不同的类别)、回归问题 (预测连续值) 或聚类问题 (将数据聚类成不同的群组) 等。

机器学习在各个领域都得到了广泛应用, 包括自然语言处理、图像和语音识别、推荐系统、金融预测、医疗诊断等。它的发展不仅为人工智能的进步提供了基础, 也为科学研究和实际问题的解决提供了强有力的工具和方法。

2、机器学习的分类:

(1) 监督学习 (Supervised Learning): 通过给定输入数据和对应的标签 (输出), 训练模型使其能够预测未来数据的标签。例如, 根据房屋的特征 (输入), 预测房价

(标签)。

代表算法：决策树、朴素贝叶斯、逻辑回归、KNN、SVM、神经网络、随机森林、AdaBoost、遗传算法；

(2) 无监督学习 (Unsupervised Learning)：在没有标签信息的情况下，通过发现数据之间的内在关系和结构，进行数据的聚类、降维或异常检测等任务。例如，对于一组消费者的购物数据，无监督学习可以帮助识别不同的购物群体。无监督学习一般是作为有监督学习的前期数据处理，功能是从原始数据中抽取必要的标签信息。

代表算法：主成分分析方法PCA等，等距映射方法、局部线性嵌入方法、拉普拉斯特征映射方法、黑塞局部线性嵌入方法、局部切空间排列方法等；

(3) 强化学习 (Reinforcement Learning)：通过与环境的互动学习，通过不断尝试行动来最大化某种奖励信号。强化学习通常在具有明确奖励和惩罚的情景中应用，例如机器人学习走路。

(4) 半监督学习 (Semi-supervised Learning) 半监督学习包含大量未标注数据和少量标注数据。主要是利用未标注中的信息，辅助标注数据，进行监督学习。

例如说上传的照片都是大量未标注数据，但会有重复的同一个人的照片，可以通过无监督学习进行分类；如果你为其中一份照片标注了信息，则可以为其他未标注的数据标注信息。大多数半监督学习算法是无监督式和监督式算法的结合，例如深度信念网络 (DBN)。它基于一种互相堆叠的无监督式组件，这个组件叫作受限玻尔兹曼机 (RBM)。

代表算法：self-training(自训练算法)、generative models生成模型、SVMs半监督支持向量机、graph-based methods图论方法、multiview learning多视角算法等；

参考资料：[人工智能导论\(6\)——机器学习\(Machine Learning\) 机器学习 machine laearning hustlei的博客-CSDN博客。](#)

三、深度学习概述

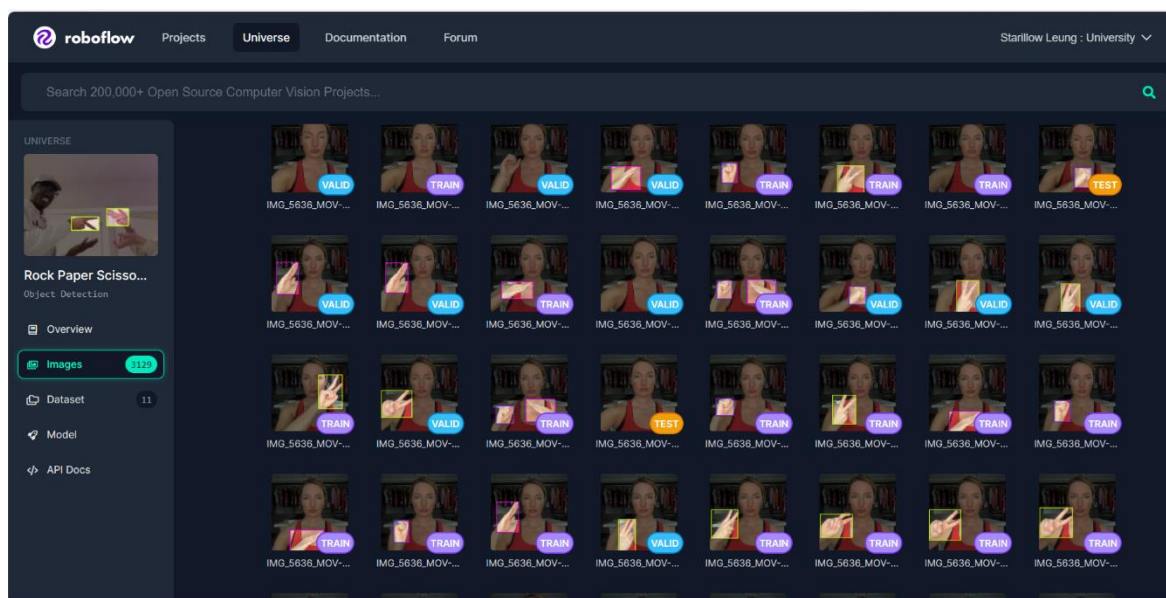
1、什么是深度学习

深度学习是一种机器学习中监督学习的技术，其核心思想是通过构建和训练神经网络模型来模拟人脑的工作方式，以解决复杂问题。与传统的机器学习方法相比，深度学习可以自动从数据中学习特征表示，而无需手动提取特征。它通过多层的神经网络结构，将数据从输入层传递到输出层，逐层提取高层抽象特征。

2、深度学习的实现过程

(1) 数据集准备：收集和准备用于训练的数据集，包括输入数据和对应的标签或目标值。数据集切分为训练集、验证集和测试集。训练集用以训练深度学习模型；验证集用以评估模型结果，进而辅助模型调参；测试集用以模型的预测。一般而言，训练集、验证集与测试集的比例为7:2:1。

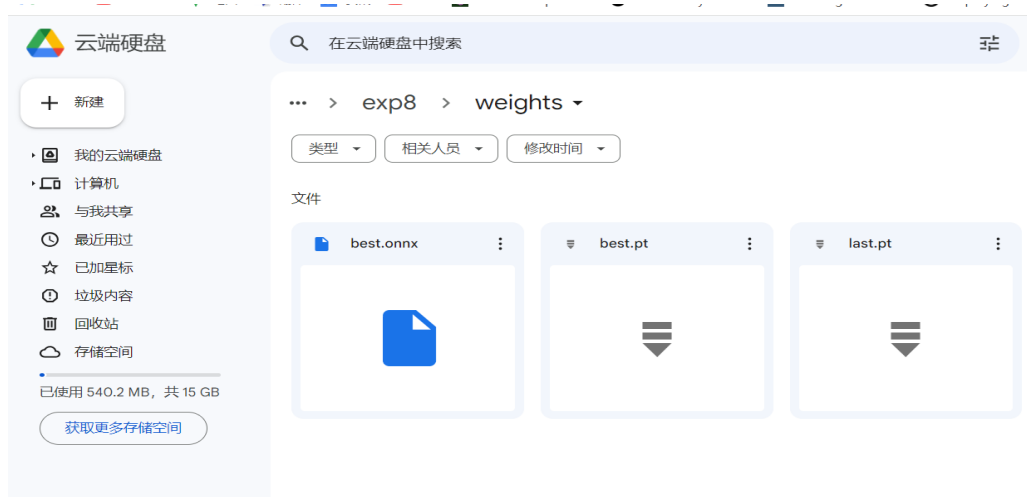
Roboflow是一个用于计算机视觉项目管理和数据预处理的平台。它提供了一系列工具和功能，以帮助开发人员更轻松地进行构建、训练和部署计算机视觉模型。下图中是Roboflow中所展示的一个已经标注好标签的数据集（石头剪刀布）。我们可以看到数据集分为了train、test、valid三个类型用于训练模型。



(2) 网络构建：选择适当的神经网络结构，如多层感知机（MLP）、卷积神经网络（CNN）或循环神经网络（RNN），并设置网络的层数和节点数。

(3) 模型训练：使用训练数据集来训练神经网络模型。这涉及到将输入数据传递给网络，计算输出结果，并通过反向传播算法来调整网络参数，以最小化预测结果与实际标签之间的差距。

训练出来的模型通常是某种特定格式的文件。



(4) 模型评估：使用测试数据集来评估训练好的模型的性能。这可以通过计算准确率、精确度、召回率等指标来衡量。

(5) 模型优化：根据评估结果，对模型进行调整和优化，以提高其性能。这可能包括调整网络结构、改变超参数或使用正则化技术等。

3、深度学习的应用

深度学习在许多领域都有广泛的应用，包括计算机视觉、自然语言处理、语音识别、推荐系统等。例如，深度学习可以用于图像分类、目标检测、图像生成等计算机视觉任务；用于文本分类、机器翻译、情感分析等自然语言处理任务；用于语音识别、语音合成等语音相关任务；以及用于个性化推荐、广告点击率预测等推荐系统任务。深度学习的强大表达能力和自动特征学习的能力使其成为解决复杂问题的有力工具。

4、人工智能，机器学习和神经网络的关系

人工智能（AI）、机器学习（Machine Learning）、深度学习（Deep Learning, DL）和神经网络（Neural Networks）之间有密切的联系和相互依赖。

首先，人工智能是一个广义的概念，指的是使计算机能够模拟和展现出人类智能的学科。它涵盖了不同的技术和方法，包括机器学习和神经网络。

机器学习是人工智能的一个重要分支，它关注使计算机从数据中学习并进行预测和决策。机器学习的核心思想是通过构建数学模型和算法，使计算机能够从大量的数据中自动提取特征和模式，并进行泛化和预测。机器学习包括监督学习、无监督学习和强化学习等不同类型的学习方式。

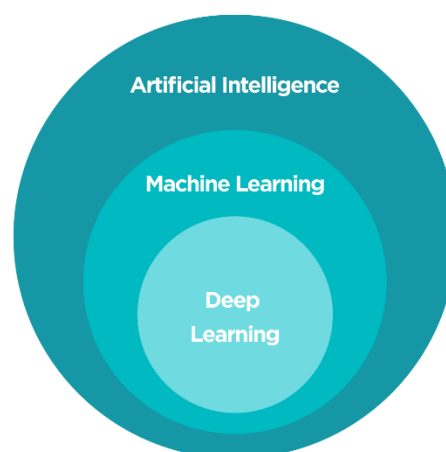
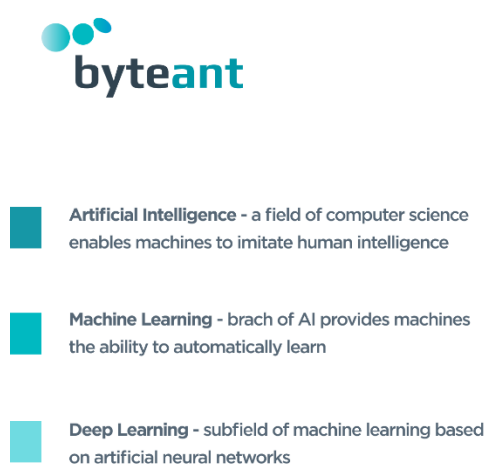
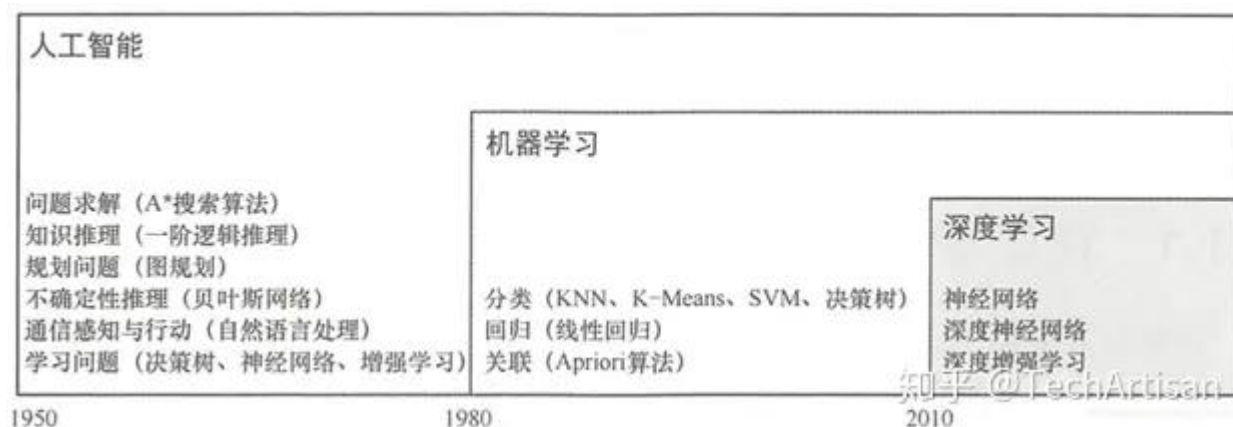
深度学习（Deep Learning, DL）是机器学习的一个特定领域，它使用多层神经网络来模拟人脑的工作方式。深度学习通过逐层提取高层抽象特征，可以自动从数据中学习表示，并在各种任务中取得了很大的成功。深度学习的核心是神经网络。

神经网络是机器学习的一个重要方法和技术，它模拟了生物神经系统的工作原理。神经网络由大量的人工神经元（节点）和它们之间的连接组成。每个神经元接收输入信号并根据权重和激活函数的计算产生输出。通过多层的神经元和权重的调整，神经网络可以学习复杂的非线性关系和特征，从而实现各种任务，如分类、回归、图像处理等。深度学习中的神经网络通常由多个层次组成，包括输入层、隐藏层和输出层。

机器学习和神经网络的关系是密切的。传统的机器学习算法可以使用不依赖神经网络的方法，如决策树、支持向量机等。然而，神经网络作为一种强大的学习模型，在许多任务中表现出了出色的性能，特别是在处理大规模、高维度的数据上。深度学习就是建立在神经网络基础上的机器学习方法，它通过多层次的神经网络进行特征提取和表示学习，使得计算机能够更好地处理复杂和抽象的数据。

总而言之，人工智能是研究如何使计算机能够模拟人类智能的学科；机器学习是人工智能的一个分支，关注如何通过从数据中学习来改进计算机的性能；深度学习是机器学习的一个特定领域，使用多层神经网络来模拟人脑的工作方式；神经网络是深度学习中的核心组件，用于学习输入和输出之间的映射关系。

参考资料：<https://zhuanlan.zhihu.com/p/86794447>。



推荐课程:

- (1) 黄海广老师机器学习课程: <https://www.icourse163.org/course/WZU-1464096179>
- (2) 吴恩达机器学习课程: [Machine Learning | Coursera](#)

任务一:

- (1) 阅读以上内容, 写一篇不少于 400 字的学习笔记, 内容包括但不限于自身对人工智能的理解、在学习过程中的感受、人工智能对你的生活产生的影响、对人工智能在当下以及未来的思考。
- (2) 通过学习以上知识和自学人工智能和机器学习相关的知识, 请选择以下十个问题中的五个进行回答, 请确保回答的内容是自身学习、思考所得。

1. 什么是机器学习？请简要解释机器学习的概念及其在实际应用中的作用。
2. 请解释一下有监督学习和无监督学习的区别，以及它们在机器学习中的应用场景。
3. 神经网络是什么？它在深度学习中起到了什么作用？请举一个实际应用案例。
4. 什么是自然语言处理（NLP）？请举例说明 NLP 在实际中的应用领域。
5. 请简要解释一下强化学习是如何工作的，并提供一个强化学习的实际应用示例。
6. 什么是过拟合（overfitting）？在机器学习中，我们如何应对过拟合问题？
7. 请简要说明卷积神经网络（CNN）在计算机视觉任务中的应用，并提供一个与之相关的实际案例。
8. 对抗性攻击是什么？它对深度学习模型有何影响？请提供一个相关的实际应用场景。
9. 什么是增强学习（reinforcement learning）？请解释增强学习中的三个关键要素以及它们的作用。
10. 解释一下聚类算法（clustering algorithm）的概念和目的，并提供一个使用聚类算法的实际案例。

任务一提交要求：

请将以上内容以 PDF 文件的格式提交至砺儒云。

学习以Python为基础的编程语言

Python是完全面向对象的编程语言，采用极简主义的设计理念，加以统一规范的交互模式，这使得Python易于学习、理解和记忆。Python在人工智能和机器学习领域中被广泛应用，学习人工智能离不开 Python 编程。在光电学院的培养方案中，Python是选择性必修的课程之一，了解和学习Python，能够帮助我们适应未来的Python课程以及更快地开始对人工智能的探索。在本次教程中，我们推荐使用VScode，anaconda。

在开始任务前，我们建议同学们先了解VScode为代表的集成开发平台（IDE）和Anaconda的关系。

1、anaconda作用

（1）Anaconda 附带了一大批常用数据科学包，它附带了 conda、Python 和 150 多个科学包及其依赖项。因此你可以立即开始处理数据。

（2）管理包

Anaconda 是在 conda（一个包管理器和环境管理器）上发展出来的。在数据分析中，你会用到很多第三方的包，而 conda（包管理器）可以很好的帮助你在计算机上安装和管理这

些包，包括安装、卸载和更新包。

(3) 管理环境

为什么要管理环境？比如你在 A 项目中用了 Python 2，而新的项目 B 老大要求使用 Python 3，而同时安装两个 Python 版本可能会造成许多混乱和错误。这时候 conda 就可以帮助你为不同的项目建立不同的运行环境。还有很多项目使用的包版本不同，比如不同的 pandas 版本，不可能同时安装两个 Numpy 版本，你要做的应该是，为每个 Numpy 版本创建一个环境，然后项目的对应环境中工作。这时候 conda 就可以帮你做到。

(4) 如果你已经安装了Anaconda，可以不需要再单独安装Python。 Anaconda是一个集成了Python和许多常用科学计算包的发行版，它会自动安装Python和其他必要的库。当你安装Anaconda时，它会在系统中创建一个独立的Python环境，并与Anaconda的库一起使用。在使用Anaconda时，你可以直接使用Anaconda自带的Python解释器来运行代码，而不需要额外安装Python。此外，Anaconda还提供了一个方便的环境管理器，可以创建和管理多个独立的Python环境，以满足不同项目的需要。

2、集成开发平台IDE的作用（VScode）

集成开发平台（Integrated Development Environment，IDE）是一种软件应用程序，旨在为开发人员提供一个全面的开发环境，包括代码编辑、调试、编译、版本控制等一系列开发任务的支持。

IDE的主要目标是提高开发人员的生产力和效率。它将多个开发工具和功能集成到一个统一的界面中，提供用于编写、测试和调试代码的工具。以下是IDE的一些主要特点和功能：

1. 代码编辑器：提供代码高亮、语法检查、自动补全等功能，以增强开发人员对代码的编写体验。
2. 编译和构建工具：包括编译器、构建工具链等，用于将源代码转换为可执行文件或库。
3. 调试器：允许开发人员逐行执行代码，查看变量值、调用堆栈等，以帮助识别和修复错误。
4. 版本控制集成：与版本控制系统（如Git）集成，方便开发人员进行代码版本管理和协作。
5. 自动化工具：提供自动化构建、部署、测试等工具，以简化繁琐的重复任务。
6. 插件和扩展：允许开发人员通过安装插件和扩展来扩展和定制IDE的功能。
7. 文档和帮助系统：提供开发文档、API参考、教程等资源，方便开发人员学习和使用相关技术。

常见的IDE包括Visual Studio、VScode、Pycharm、IntelliJ IDEA等。它们针对不同的编程语言和开发平台提供了特定的功能和工具。通过使用IDE，我们可以更加高效地进行软件开发，简化开发流程，提高代码质量和可维护性。

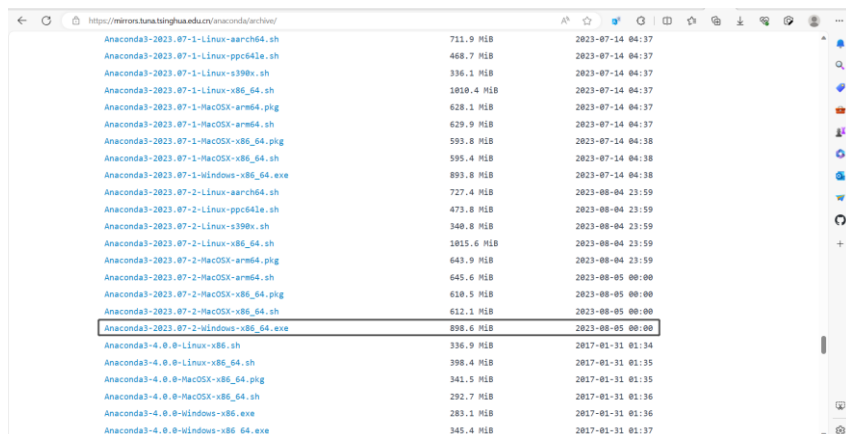
3、参考步骤：

(1) 下载VSCode：

打开浏览器，访问VS Code官方网站（<https://code.visualstudio.com/>）。点击“Download”按钮下载适用于Windows的安装程序。运行安装程序，并按照提示完成安装。

（2）安装Anaconda：

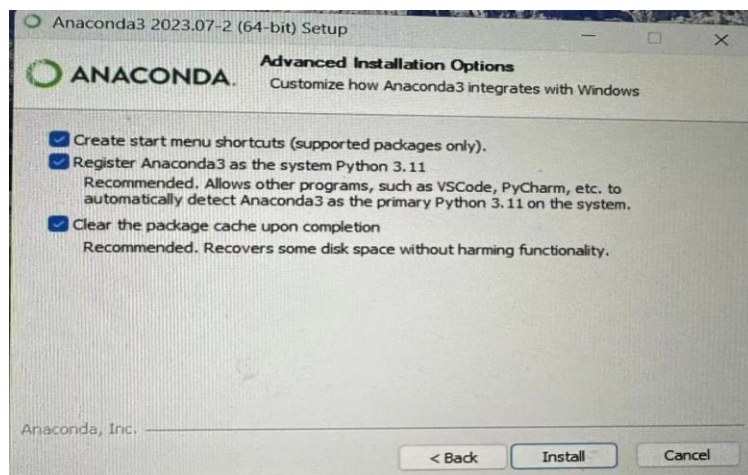
在[清华大学开源软件镜像站](https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/)（推荐）或Anaconda官方网站安装。详见<https://zhuanlan.zhihu.com/p/75717350>。在页面中，找到适用于Windows的Anaconda版本，并点击“Download”按钮下载安装程序。运行安装程序，并按照提示完成安装。注意，安装过程中可能会出现一些选项让你选择，默认选项即可。

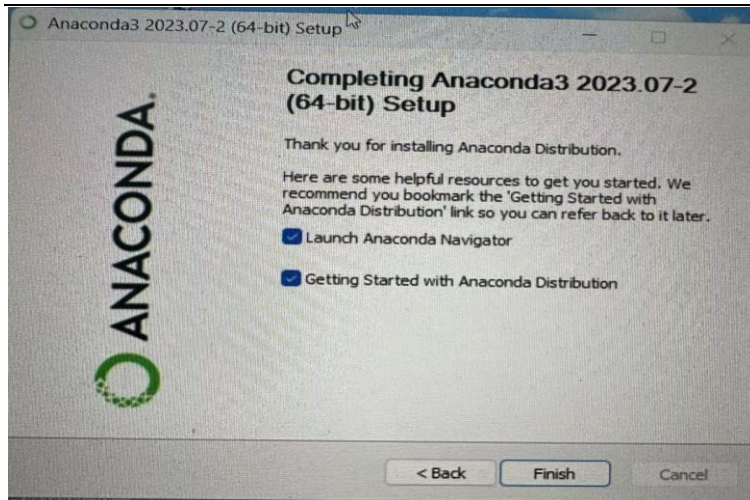


Anaconda3-2023.07-1-Linux-aarch64.sh	711.9 MiB	2023-07-14 04:37
Anaconda3-2023.07-1-Linux-ppc64le.sh	468.7 MiB	2023-07-14 04:37
Anaconda3-2023.07-1-Linux-s390x.sh	336.1 MiB	2023-07-14 04:37
Anaconda3-2023.07-1-Linux-x86_64.sh	1810.4 MiB	2023-07-14 04:37
Anaconda3-2023.07-1-MacOSX-arm64.pkg	628.1 MiB	2023-07-14 04:37
Anaconda3-2023.07-1-MacOSX-arm64.sh	629.9 MiB	2023-07-14 04:37
Anaconda3-2023.07-1-MacOSX-x86_64.pkg	593.8 MiB	2023-07-14 04:38
Anaconda3-2023.07-1-MacOSX-x86_64.sh	595.4 MiB	2023-07-14 04:38
Anaconda3-2023.07-2-Windows-x86_64.exe	893.8 MiB	2023-07-14 04:38
Anaconda3-2023.07-2-Linux-aarch64.sh	727.4 MiB	2023-08-04 23:59
Anaconda3-2023.07-2-Linux-ppc64le.sh	473.8 MiB	2023-08-04 23:59
Anaconda3-2023.07-2-Linux-s390x.sh	340.8 MiB	2023-08-04 23:59
Anaconda3-2023.07-2-Linux-x86_64.sh	1815.6 MiB	2023-08-04 23:59
Anaconda3-2023.07-2-MacOSX-arm64.pkg	643.9 MiB	2023-08-04 23:59
Anaconda3-2023.07-2-MacOSX-arm64.sh	645.6 MiB	2023-08-05 00:00
Anaconda3-2023.07-2-MacOSX-x86_64.pkg	610.5 MiB	2023-08-05 00:00
Anaconda3-2023.07-2-MacOSX-x86_64.sh	612.1 MiB	2023-08-05 00:00
Anaconda3-2023.07-2-Windows-x86_64.exe	898.6 MiB	2023-08-05 00:00
Anaconda3-4.0-Linux-x86.sh	336.9 MiB	2017-01-31 01:34
Anaconda3-4.0-Linux-x86_64.sh	398.4 MiB	2017-01-31 01:35
Anaconda3-4.0-MacOSX-x86_64.pkg	341.5 MiB	2017-01-31 01:35
Anaconda3-4.0-MacOSX-x86_64.sh	292.7 MiB	2017-01-31 01:36
Anaconda3-4.0-Windows-x86.exe	283.1 MiB	2017-01-31 01:36
Anaconda3-4.0-Windows-x86_64.exe	345.4 MiB	2017-01-31 01:37

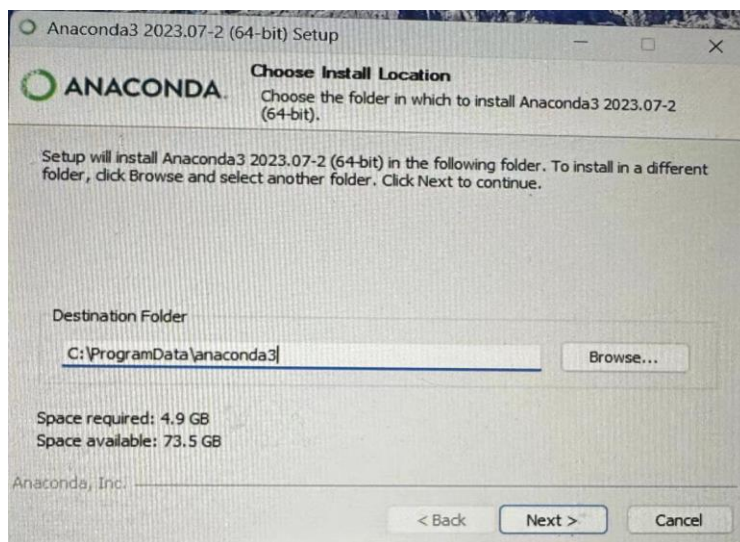
提示：

a. 建议勾选以下内容

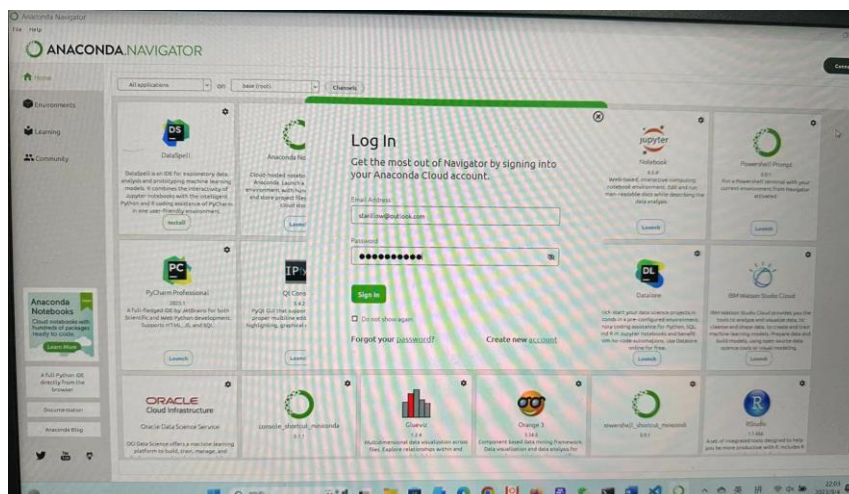




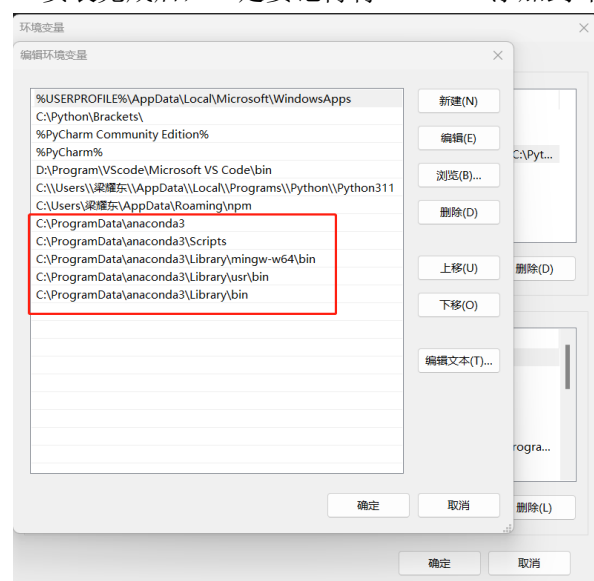
b. 建议安装在C盘



c. 安装完成后会自动导向anaconda的官网，需要在官网用邮箱注册一个账号，用于在anaconda navigator登录。

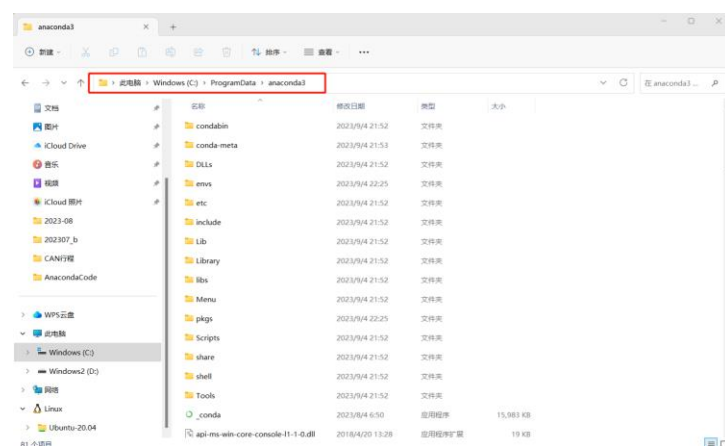


d. 安装完成后，一定要记得将anaconda添加到环境变量



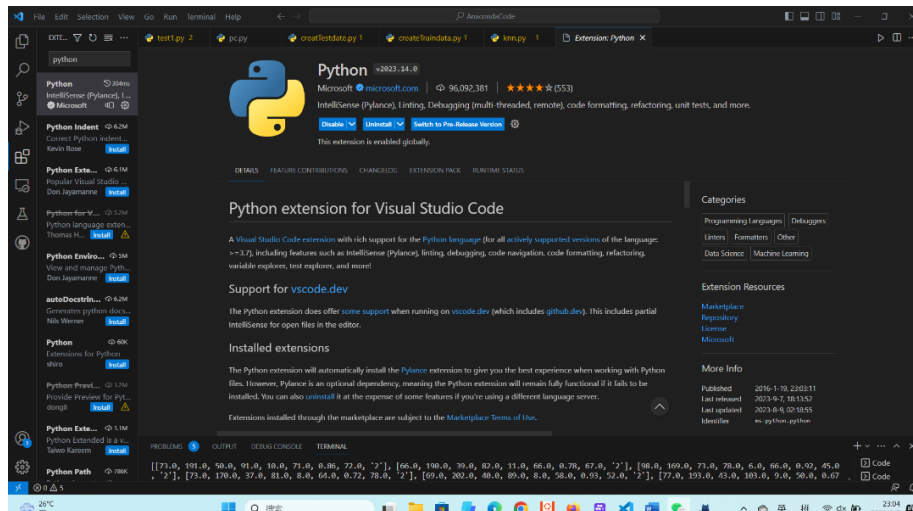
(参考)

具体安装路径可在文件夹中查询，点击红色框可得到路径，复制路径到环境变量中。



(3) 配置VScode的Python插件:

打开VS Code，在左侧的侧边栏中点击扩展图标。在搜索栏中输入“Python”，找到Microsoft官方提供的Python扩展，并点击“Install”按钮进行安装。安装完成后，点击“Reload”重新加载VS Code。



（4）配置（选择）Anaconda环境：

在VS Code的顶部菜单栏中，点击“View” -> “Command Palette”（或使用快捷键Ctrl+Shift+P）。在命令面板中，输入“Python: Select Interpreter”并选择该命令。在弹出的列表中，选择Anaconda的Python解释器，一般为类似于“Python 3.x.x 64-bit（‘base’:conda）”的选项。确认选择后，VS Code会在底部状态栏中显示所选解释器。

4、学习Python基础用法

推荐大家学习 b 站小甲鱼的 python 课程。小甲鱼的课程简单易懂，并且进度不慢，覆盖面广，初学者可粗略地阅览 python 的大概语法。并且小甲鱼也有 C 语言与 Web前端的课程。

另外推荐菜鸟教程（<https://www.runoob.com/python/python-tutorial.html>）。这是 Python 使用说明书，讲解全面，像字典一样即查即用。

在线课程推荐：<https://www.icourse163.org/course/BIT-268001>

任务二：

1、通过以上资料，完成Python环境的搭建。

2、请从以下六道Python基础题目中选择三道进行编程练习：

- （1）编程实现，让用户输入三个整数，要求打印输出最大的和最小的数字。
- （2）编程实现，使用 while 循环实现输出 1-100 内的所有奇数。
- （3）编程实现，模拟出租车计价器，具体收费标准如下3公里以内收费13元，超出3公里以外，每公里基本单价2.3元/公里，空驶费：超过15公里后每公里加收1.15元空驶费
- （4）编程实现，让用户输入10个数字，要求输出这10个数字，并输出10个数字中最大的和最小的数字

- (5) 将字符串abc123转换为321cba
- (6) 输入一行字符，分别统计出其中英文字母、空格、数字和其它字符的个数。

任务二提交要求：

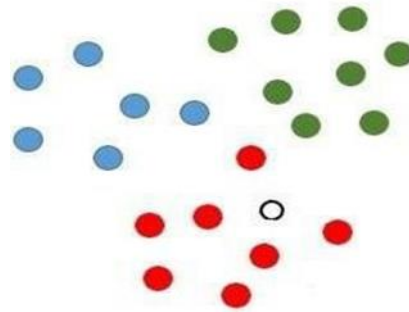
请在运行的代码中以注释的形式加入自己的姓名和学号，将代码截图以及运行结果的截图放在一个文件中，提交方式为 PDF 文件格式。

进阶任务（选做）

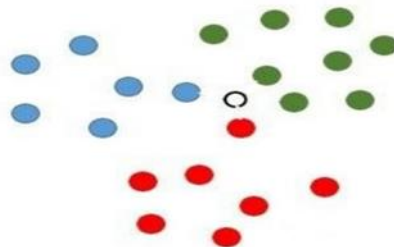
在进阶任务中，请同学们学习k-近邻算法，了解其含义、逻辑以及应用的方式。

1、k-近邻算法介绍：

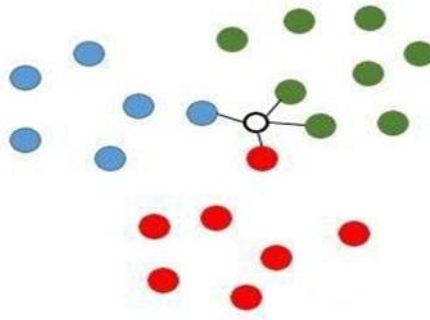
在分类问题中，通常是给出一个未知数据，要求人工智能判断未知数据的类别。假设我们现在有红、绿、蓝三种颜色的点，分布在二维空间中。一个新的数据点如下图所示，那么，该数据点大概率被分类为哪个类别？



很显然，应该被分类为红色类别。再看下面一张图，新数据点应该是哪个类别？



貌似不太好判断，但我们会认为这可能是绿色类别，因为有两个绿色的数据点比较靠近。



这就是 k-近邻算法的核心思想：以新数据点为中心，选取最“近”的 k 个已知数据点，在这 k 个数据点找到最多的类别。如第二个例子，取 $k=4$ ，则可找到最近的 4 个数据点，其中一个蓝色、一个红色、两个绿色，所以机器学习判断这是个绿色类别。

2、“距离”的定义？

在二维空间中，设两点 $A = (x_1, y_1)$ ， $B = (x_2, y_2)$ ，则它们的距离表示为：

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

三维空间也很容易解释，设两点 $A = (x_1, y_1, z_1)$ ， $B = (x_2, y_2, z_2)$ ：而通常数据的特征不止三个例如给定历史房价要求我们预测一座新房子的价格，我们需要知道其面积、楼层、房间数、地理位置等等，这些叫做数据的“特征”，也叫“标签”或“属性”。

假设给定数据集中有 n 个特征，则两点的距离可表示为

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_n - y_n)^2} = \left(\sum_{i=1}^n |x_i - y_i|^2 \right)^{\frac{1}{2}}$$

这是最常用的、字面意义的距离，也叫欧几里得距离。

事实上，“距离”的表述多种多样，除了欧几里得距离之外，还有很多抽象的“距离距离”。定义下的结果可能不同。更一般的距离有 L_p 距离。

$$L_p(x, y) = \left(\sum_{i=1}^n |x_i - y_i|^p \right)^{\frac{1}{p}}$$

下面的例子说明，由不同的距离度量所确定的最近邻点是不同的。

例 已知二维空间中的 3 个点 $x_1 = (1, 1)^T$, $x_2 = (5, 1)^T$, $x_3 = (4, 4)^T$ ，试求在 p 取不同值时， p 距离下 x_1 的最近邻点。

解 因为 x_1 与 x_2 只有第二维上值不同，所以 p 取任意值时， $L_p(x_1, x_2) = 4$

$$L_1(x_1, x_3) = 6, L_2(x_1, x_3) = 4.24, L_3(x_1, x_3) = 3.78, L_4(x_1, x_3) = 3.57$$

于是得到：p 等于 1 或 2 时， x_2 是 x_1 的最近邻点；p 大于等于 3 时， x_3 是 x_1 的最近邻点。

3、机器学习怎样运用模型？

对于任何模型来说，运用模型不外乎六步。

Step1: 收集数据：可以使用任何方法。

Step2: 准备数据：计算距离所需要的数值，最好是结构化的数据格式。

Step3: 分析数据：可以使用任何方法。

Step4: 训练算法：此步骤不适用于 k-近邻算法。

Step5: 测试算法：计算错误率。

Step6: 使用算法：首先需要输入样本数据和结构化的输出结果，然后运行 k-近邻算法判定输入数据分别属于哪个分类，最后应用对计算出的分类执行后续的处理。

如果得到的错误率较高，那么表明该模型不适用于待测数据集。

4、k 的取值应该怎么确定？

k 值的选择会对 k 近邻法的结果产生重大影响。如果选择较小的 k 值，就相当于用较小的邻域中的训练实例进行预测，“学习”的近似误差会减小，只有与输入实例较近的（相似的）训练实例才会对预测结果起作用，但缺点是“学习”的估计误差会增大，预测结果会对近邻的实例点非常敏感。如果邻近的实例点恰巧是噪声，预测就会出错。换句话说，k 值的减小就意味着整体模型变得复杂，容易发生过拟合。

如果选择较大的 k 值，就相当于用较大邻域中的训练实例进行预测。其优点是可以减少学习的估计误差。但缺点是学习的近似误差会增大。这时与输入实例较远的（不相似的）训练实例也会对预测起作用，使预测发生错误。k 值的增大就意味着整体的模型变得简单。

如果 $k=N$ （样本总数），那么无论输入实例是什么，都将简单地预测它属于在训练实例中最多的类，这时，模型过于简单，完全忽略训练实例中的大量有用信息，是不可取的。在应用中，k 值一般取一个比较小的数值。通常采用交叉验证法来选取最优的 k 值。

5、低维嵌入

以上讨论是基于一个重要假设：任意测试样本 x 附近任意小的 δ 范围内总能找到一个训练样本，即训练样本的采样密度足够大，或称为“密采样”。然而，这个假设在现实任务中通常很难满足，例如若 $\delta = 0.001$ ，仅考虑单个属性，则仅需 1000 个样本点平均分布在归一化后的属性取值范围内，即可使得任意测试样本在其附近 0.001 距离范围内总能找到一个训练样本。然而，这仅是属性维数为 1 的情形，若有更多的属性，则情况会发生显著变化。例如假定属性维数为 20，若要求样本满足密采样条件，则至少需 10320 个样本。现实应用中属性维数经常成千上万，要满足密采样条件所需的样本数目是无法达到的天文数字。此外，许多学习方法都涉及距离计算而高维空间会给距离计算带来很大的麻烦。

事实上，在高维情形下出现的数据样本稀疏、距离计算困难等问题，是所有机器学习方法共同面临的严重障碍，被称为“维数灾难”。缓解维数灾难的一个重要途径是降维，亦称“维数约简”，即通过某种数学变换将原始高维属性空间转变为一个低维“子空间”，在这个子空间中样本密度大幅提高，距离计算也变得更为容易。为什么能进行降维？这是因为在很多时候，人们观测或收集到的数据样本虽是高维的，但与学习任务密切相关的也许仅是某个低维分布，即高维空间中的一个低维“嵌入”。下图给出了一个直观的例

子，原始高维空间中的样本点，在这个低维嵌入子空间中更容易进行学习。

若要求原始空间中样本之间的距离在低维空间中的距离在低维空间中得以保持，则可用“多维缩放”这样一种经典的降维算法。因公式过于复杂在此不再展开，只需了解这一概念即可

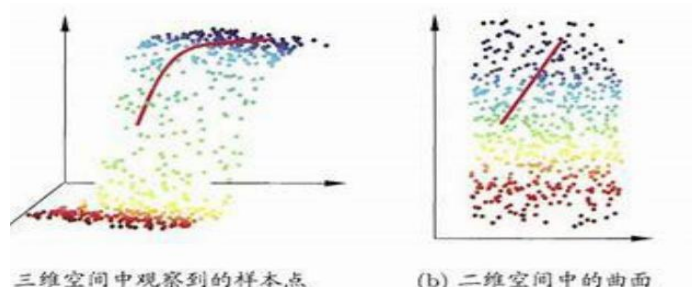


图 10.2 低维嵌入示意图

6、拓展资料

(1) k-近邻法的实现需要考虑如何快速搜索 k 个最近邻点。kd 树是一种便于对 k 维空间中的数据进行快速检索的数据结构。kd 树是二叉树，表示对 k 维空间的一个划分，其每个结点对应于 k 维空间划分中的一个超矩形区域。利用 kd 树可以省去对大部分数据点的搜索，从而减少搜索的计算量。有兴趣的同学不妨查阅相关资料，了解 kd 树的构造方法与原理。

(2) k-近邻算法不仅可用于分类，还可以用于回归。这种回归的方法叫“k-近邻回归”。如果你好奇该算法怎样进行回归，不妨一查。

(3) 如果你已经搭建好了 Python 的环境，也可尝试运行示例程序，看看 Python 进行机器学习的步骤。

7、推荐参考资料：

一文搞懂k近邻(k-NN)算法(一)：<https://zhuanlan.zhihu.com/p/25994179>

史上最全面K近邻算法/KNN算法详解+python实现：

<https://zhuanlan.zhihu.com/p/341572059>

进阶任务一：

通过以上的内容，你应该了解了 k-近邻算法的大致原理。但实践是检验真理的唯一标准，下面请你稍作整理，尝试推理运用 k-近邻算法的步骤，利用编程块的思想，给出 k-近邻算法的伪代码（不需要进行低维嵌入）。

下面给出伪代码的示例。可以直接复制下面的大体框架进行编辑，也可以自行编辑。

输入：训练集 $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$

一个未知点 (x_t, y_t)

过程：

```
1. 分析数据，通过画图等方法大概浏览数据的分布
2. 将数据转化为便于计算的数字
3. .... (这里你需要编辑)
4. for .... do(如果有需要的话)
5. ....(这里可以不止一行)
6. end for
7. if ....(如果有需要的话) then
8. ....(这里可以不止一行)
9. end if(如果不需条件判断，则删除整个 if 结构，下同)
10. Repeat {
11. ....
12. }
13. ....
14. 得到未知点的预测的分类
```

输出：未知点的预测的分类

伪代码没有统一的格式，可使用任意中文与英文表述语言（比如 if, for.....do 等），自由发挥，只需保证步骤清晰、严谨。

提交要求：

**请在运行的代码中以注释的形式加入自己的姓名和学号，
将代码截图以及运行结果的截图放在一个文件中，提交
方式为 PDF 文件的格式。**

进阶任务二：KNN算法预测城市空气质量

(1) 了解k近邻算法，按照相应的步骤运行代码，理解运行的代码以及其中的函数的作用。

(2) 运行以下示例代码，更改k的值，得到最优的预测结果。

任务参考文章：

https://blog.csdn.net/SeizeeveryDay/article/details/112792514?fromshare=blogdetail&sharetype=blogdetail&sharerId=112792514&sharerrefer=PC&sharesource=m0_73789500&sharefrom=from_link

任务步骤：

1、获取数据。数据来源：<http://www.tianqihoubao.com/aqi/chengdu-201901.html>。创建anaconda工作环境，创建py文件，运行代码（右上角三角形）来获取2019年特定地区的

天气数据（代码可在参考文章中获取），通过修改地区拼音，得到两个不同地区的数据（csv文件）。

```

main2.py  main3.py  model.py 7  pc.py  X
GuangXie > 上学期 > pc.py > ...
1  # -*- coding: UTF-8 -*-
2  import pandas as pd
3  import logging
4
5  logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s: %(message)s')
6
7  for page in range(1, 13): # 12个月
8      if page < 10:
9          url = f'http://www.tianqihoubao.com/aqi/shanghai-20190{page}.html'
10         df = pd.read_html(url, encoding='gbk')[0]
11         if page == 1:
12             df.to_csv('2019年上海空气质量数据.csv', mode='a+', index=False, header=False)
13         else:
14             df.iloc[1:,:].to_csv('2019年上海空气质量数据.csv', mode='a+', index=False, header=False)
15     else:
16         url = f'http://www.tianqihoubao.com/aqi/guangzhou-2019{page}.html'
17         df = pd.read_html(url, encoding='gbk')[0]
18         df.iloc[1:,:].to_csv('2019年上海空气质量数据.csv', mode='a+', index=False, header=False)
19
20     logging.info(f'{page}月空气质量数据下载完成! ')

```

2、生成测试集和训练集。将一个城市数据选定为测试集，另一个选定为训练集。创建两个py文件，分别用于生成测试集和训练集。运行代码后可分别得到一个txt文件。

```

File Edit Selection View Go Run Terminal Help AnacondaCode
EXPLORER
2019年上海空气质量
2019年广州空气质量
createTraindata.py
data.csv
knn.py
pc.py
test.txt
test.py
train.txt

createTraindata.py 1 X
1 import pandas as pd
2
3 # 自定义其他几个城市空气质量数据作为训练集
4 df = pd.read_csv('2019年上海空气质量数据.csv', encoding='utf-8')
5 # 取空气质量 Aqi指数 当天Aqi排名 PM2.5 ... 等数据
6 # settingsWithKeyWarning: A value is trying to be set on a copy of
7 df1 = df[['AQI指数', '当天AQI排名', 'PM2.5', 'PM10', 'SO2', 'NO2', '']]
8
9 air_quality = []
10 # print(df1['AQI指数'].value_counts())
11 # 质量等级列数据为字符串 转为数字标识
12 for i in df1['AQI指数']:
13     if i == "优":
14         air_quality.append('1')
15     elif i == "良":
16         air_quality.append('2')
17     elif i == "轻度污染":
18         air_quality.append('3')
19     elif i == "中度污染":
20         air_quality.append('4')
21     elif i == "重度污染":
22         air_quality.append('5')
23     elif i == "严重污染":
24         air_quality.append('6')
25
26 print(air_quality)
27 df1['空气质量'] = air_quality
28 # 将数据写入追加写入到train.txt
29 # print(df1.values, type(df1.values)) # <class 'numpy.ndarray'>
30 with open('train.txt', 'a+') as f:
31     for x in df1.values:
32         print(x)
33         s = ','.join([str(i) for i in x])
34         # print(s, type(s))
35         f.write(s + '\n')
36
createTestdata.py 1 X
1 import pandas as pd
2
3 # 将2019年广州空气质量数据作为测试集
4 df = pd.read_csv('2019年广州空气质量数据.csv')
5 # 取空气质量 Aqi指数 当天Aqi排名 PM2.5 ... 等数据
6 # settingsWithKeyWarning: A value is trying to be set on a copy of
7 df1 = df[['AQI指数', '当天AQI排名', 'PM2.5', 'PM10', 'SO2', 'NO2', '']]
8
9 air_quality = []
10 # print(df1['AQI指数'].value_counts())
11 # 质量等级列数据为字符串 转为数字标识 便于判断预测
12 for i in df1['AQI指数']:
13     if i == "优":
14         air_quality.append('1')
15     elif i == "良":
16         air_quality.append('2')
17     elif i == "轻度污染":
18         air_quality.append('3')
19     elif i == "中度污染":
20         air_quality.append('4')
21     elif i == "重度污染":
22         air_quality.append('5')
23     elif i == "严重污染":
24         air_quality.append('6')
25
26 print(air_quality)
27 df1['空气质量'] = air_quality
28 # 将数据写入到test.txt
29 # print(df1.values, type(df1.values)) # <class 'numpy.ndarray'>
30 with open('test.txt', 'w') as f:
31     for x in df1.values:
32         print(x)
33         s = ','.join([str(i) for i in x])
34         # print(s, type(s))
35         f.write(s + '\n')
36
Ln 14, Col 32 Spaces: 4 UTF-8 CRLF Python 3.11.1 64 bit

```

3、实现KNN算法，完成代码。以下是代码主体，函数中的函数主体可在参考文章中得到。

```

import pandas as pd
import csv
import operator
import math

class MyClassifier:
    def __init__(self):
        self.training_set = [] # 训练集

```

```
self.test_set = []          # 测试集
def read_dataset(self,filename1, filename2, trainingSet, testSet):
    函数主体
def calculateDistance(self,testdata, traindata, length): # 计算距离
    函数主体
def getNeighbors(self, trainingSet, test_instance, k): # 返回最近的 k 个
边距
    函数主体
def getResponse(self,neighbors): # 根据少数服从多数，决定归类到哪一类
    函数主体
def getAccuracy(self,test_set, predictions):
    函数主体
def run(self):
    函数主体
# 创建你的分类器实例
my_classifier = MyClassifier()

trainingSet =[]
testSet=[]

# 指定训练和测试数据文件名
filename2 = '2019 年上海空气质量数据.csv'
filename1 = '2019 年广州空气质量数据.csv'

filename2 = 'train.txt'
filename1 = 'test.txt'

# 调用 read_dataset 等函数来准备数据集
my_classifier.read_dataset(filename1,filename2,trainingSet,testSet)

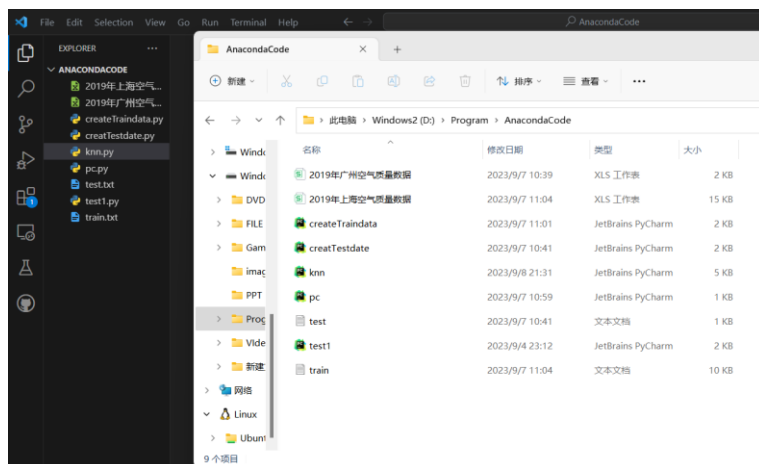
# 调用 run 函数来运行你的代码
my_classifier.run()
```

提醒：

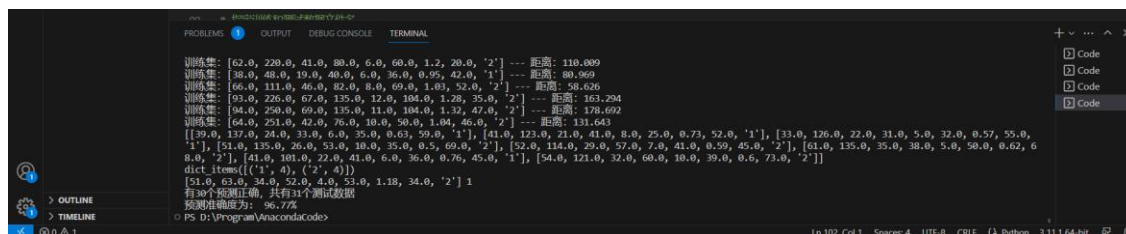
(1) 在run（）函数，读取数据集的函数中，第一二个参数是txt文件当前路径，若文件均在同一文件夹，可直接直接调用。

```
def run(self):
    training_set = []      # 训练集
    test_set = []         # 测试集
    self.read_dataset('test.txt', 'train.txt', training_set, test_set)
    print('Train set: ' + str(len(training_set)))
    print('Test set: ' + str(len(test_set)))
```

(2) 任务所需文件概览。



(3) 运行代码后即可得到预测数据



提交要求:

**请在运行的代码中以注释的形式加入自己的姓名和学号，
将代码截图以及运行结果的截图放在一个文件中，提交
方式为 PDF 文件的格式。**