

上海大学1

毕业设计（论文）

题目：基于证据学习的决策树的可视化实现

*装订线*

学 院：中欧工程技术学院

专 业：信息工程

学 号：18124666

学生姓名：诸逸冰

指导教师：陈伦德

起讫日期：2022.3.1-2022.6.15

目录

[摘要 III](#_Toc104558580)

[ABSTRACT IV](#_Toc104558581)

[第1章 绪论 1](#_Toc104558582)

[§1.1 证据决策树可视化的背景及意义 1](#_Toc104558583)

[§1.2 证据决策树的研究现状及存在的问题 2](#_Toc104558584)

[§1.2.1 研究现状 2](#_Toc104558585)

[§1.2.2 存在的问题 2](#_Toc104558586)

[§1.3 本文研究内容及目标 2](#_Toc104558587)

[§1.3.1 研究内容 2](#_Toc104558588)

[§1.3.2 研究目标 3](#_Toc104558589)

[§1.4 本文组织结构 3](#_Toc104558590)

[第2章 证据决策树相关理论 4](#_Toc104558591)

[§2.1 证据理论 4](#_Toc104558592)

[§2.1.1 识别框架 4](#_Toc104558593)

[§2.1.2 质量函数 4](#_Toc104558594)

[§2.1.3 信任函数 5](#_Toc104558595)

[§2.1.4 似然度函数 5](#_Toc104558596)

[§2.1.5 众信度函数 6](#_Toc104558597)

[§2.1.6 转换函数 6](#_Toc104558598)

[§2.2 决策树 6](#_Toc104558599)

[§2.2.1 构造方式 7](#_Toc104558600)

[§2.2.2 常见决策树算法 7](#_Toc104558601)

[§2.3 证据决策树 7](#_Toc104558602)

[§2.3.1 构造方式 8](#_Toc104558603)

[§2.3.2 证据距离的计算 8](#_Toc104558604)

[§2.3.3 最佳区分属性的选择 9](#_Toc104558605)

[§2.3.4 停止规则和叶结点处理 9](#_Toc104558606)

[§2.3.5 证据融合规则 10](#_Toc104558607)

[§2.3.6 节点的平均相似性 10](#_Toc104558608)

[§2.4 本章小结 11](#_Toc104558609)

[第3章 可视化开发环境及证据决策树算法的实现 12](#_Toc104558610)

[§3.1 可视化技术的概念 12](#_Toc104558611)

[§3.2 Manim库简介 12](#_Toc104558612)

[§3.2.1 运行方式 12](#_Toc104558613)

[§3.2.2 渲染方式 14](#_Toc104558614)

[§3.2.3 Manim 主要目录结构 15](#_Toc104558615)

[§3.2.4 基于Manim的可视化案例 17](#_Toc104558616)

[§3.3 证据决策树模型的搭建 21](#_Toc104558617)

[§3.3.1 质量函数的实现 21](#_Toc104558618)

[§3.3.2 数据集定义方式 21](#_Toc104558619)

[§3.3.3 证据决策树的基本功能函数 22](#_Toc104558620)

[§3.3.4 证据决策树的构造逻辑 24](#_Toc104558621)

[§3.3.5 证据决策树的预测 25](#_Toc104558622)

[§3.4 本章小结 26](#_Toc104558623)

[第4章 证据决策树可视化具体实现 27](#_Toc104558624)

[§4.1 主要核心概念的可视化 27](#_Toc104558625)

[§4.1.1 识别框架的动画 27](#_Toc104558626)

[§4.1.2 信任函数与似然函数的动画 28](#_Toc104558627)

[§4.1.3 树结构的动画 29](#_Toc104558628)

[§4.1.4 机器学习中的决策树的动画 31](#_Toc104558629)

[§4.1.5 证据决策树的动画 32](#_Toc104558630)

[§4.1.6 叶节点处理的动画 32](#_Toc104558631)

[§4.2 动画整体设计逻辑 33](#_Toc104558632)

[§4.2.1 动画渲染过程 33](#_Toc104558633)

[§4.3 整体效果展示 35](#_Toc104558634)

[§4.3.1 程序文件框架 35](#_Toc104558635)

[§4.3.2 程序使用说明及代码运行结果展示 35](#_Toc104558636)

[§4.3.3 提供的 API 函数接口 38](#_Toc104558637)

[§4.4 本章小结 40](#_Toc104558638)

[第5章 总结与展望 41](#_Toc104558639)

[§5.1 本文总结 41](#_Toc104558640)

[§5.1.1 本文的主要工作 41](#_Toc104558641)

[§5.1.2 本文的主要创新点 41](#_Toc104558642)

[§5.2 展望 41](#_Toc104558643)

[§5.2.1 本文的应用场景 42](#_Toc104558644)

[§5.2.2 本文的不足及改进措施 42](#_Toc104558645)

[致谢 43](#_Toc104558646)

[参考文献 44](#_Toc104558647)

基于证据学习的决策树的可视化实现

# 摘要

在人工智能领域中，基于信念函数的证据学习因其在处理不确定信息方面具有良好表现，而在近年来越来越受到关注。然而目前该领域相关的入门资料较少，阻碍了初学者进入该科研领域。同时，因其基于信念函数的理论有别于传统概率论的数学表示，且没有成熟的可视化工具，使得初学者无法获得一个直观视觉感受，加大了入门难度。

因此，本文将选取基于证据学习的决策树的可视化入手，在完成对证据决策树可视化的过程中，兼顾证据学习的基础理论介绍，编写以视频和可交互动画为载体的可视化框架。该框架基于开源数学可视化工具Manim，结合我们自己实现的证据决策树算法，做到（1）能结合实际数据集，完整展示证据决策树模型的训练、预测等过程，说明证据决策树算法的整体逻辑思路；（2）将证据决策树背后复杂的数学表示使用简单明了的动画呈现；（3）提供了决策树和证据决策树的API接口，可生成两种算法运行的实时展示。

关键词：证据决策树，可视化，证据理论，信念函数

Visual realization of decision tree based on evidence learning

# ABSTRACT

In the field of artificial intelligence, evidence learning based on belief functions has received more and more attention in recent years because of its good performance in dealing with uncertain information. However, there are few introductory materials related to this field at present, which hinders beginners from entering this field of scientific research. At the same time, because the theory based on belief function is different from the mathematical representation of traditional probability theory, and there is no mature visualization tool, it makes it impossible for beginners to obtain an intuitive visual experience, which increases the difficulty of entry.

Therefore, this paper will start with the visualization of the decision tree based on evidence learning. In the process of completing the visualization of the evidence decision tree, the basic theory of evidence learning will be taken into account, and a visualization framework based on video and interactive animation will be written. The framework is based on the open source mathematical visualization tool Manim, combined with our own implementation of the evidence decision tree algorithm, so that (1) it can combine the actual data set to fully display the training, prediction and other processes of the evidence decision tree model, and illustrate the overall evidence decision tree algorithm and logical ideas; (2) The complex mathematical representation behind the evidence decision tree is presented with simple and clear animation; (3) The API interface of the decision tree and the evidence decision tree is provided, which can generate a real-time display of the operation of the two algorithms.

**Keywords:** Evidence decision tree, Visualization, Evidence theory, Belief function

# 绪论

本章主要介绍了基于证据学习的决策树的由来和对处理机器学习中信息不确定性的作用。描述了对证据学习的决策树进行可视化的意义及用途，进而引出本文对基于证据学习的决策树的可视化的动画的设计的研究内容及目标。

## 证据决策树可视化的背景及意义

随着科技的进步，人们在日常生活中产生了数量庞大的数据，因而促使了人工智能领域的发展。特别是其中机器学习领域的进步，使得我们能更高效的利用海量数据去创造更多更好的价值。机器学习中，分类问题是一个常见任务，其目的是基于标记实例的学习集来预测一个新样本的类别。这之中，决策树又是机器学习方法中一类常用的分类算法。由于其易于理解的结构和类似人类处理问题的机制，往往在对准确数据处理和分析中具有良好的效果和直观感受。但是，多数情况下，现实中的数据往往都是具有一定的不确定性和模糊性，我们通常无法获取到完全精准的数据，或者成本过大。这种困难往往会导致错误的分类，无法实现给新样本一个精确的分类结果。例如，当数据的特征属性不包含足够的信息来识别一个精确的类，并且多个类具有相似的概率时，就会发生分歧[17]。在面对这一类情况的时候，决策树算法的性能就受到了极大的影响。针对数据的不确定性这一现实性问题，学者们提出了许多方案，其中基于信任函数的证据学习理论的方案，凭借其对数据观测值、信息源不可靠等多种复杂不确定性信息，可以进行灵活真实地描述并且添加不确定描述的特点，近年来而受到了广泛的关注，成为了机器学习中处理不确定性数据的理想手段。

证据学习，也称信念函数理论，是不确定性推理的框架之一，其在处理信息源不确定的情况方面具有明显优势。但由于该领域仍然属于新兴领域，相关的资料及基础工具较少，且证据学习理论有别于传统概率和不精确理论，具有复杂的数学表示，使得对该领域感兴趣的人望而却步，不利于该领域的推广发展。

因此，本文希望对证据学习的抽象概念可视化、程序执行可视化和算法可视化三方面，以视频动图和可交互动画的方式，对证据学习的算法进行一个简单的展示，帮助初学者对该领域拥有一个更加直观直接的感受，帮助树立对该领域基础大致知识框架。也可用于证据学习方向的教学，让科研工作者或学生从代码层面加深对证据学习的理解。由于决策树算法具有清晰的结构，具有良好的阐述性，因此本文选取基于证据学习的决策树这一角度入手，通过如何从传统决策树算法到证据决策树算法过程的可视化实现，来进行展示证据学习领域基础以及证据决策树构建的相关知识体系，同时提供针对决策树和证据决策树的两个动画生成的函数接口。

## 证据决策树的研究现状及存在的问题

证据理论也或D-S理论，最初由Arthur Dempster 于 1967 年，在统计推断的背景下引入[1]，后来由Glen Shafer发展成为一个建模认知不确定性的一般框架——一种数据证据理论[2]。后经过多位学者的研究和理论建设，使得证据学习在人工智能领域取得了实际应用和发展。但其相关配套的可视化工具及资料较少。

### 研究现状

对于理论和算法的可视化有助于提升初学者对领域内知识的获取的难易度，好的可视化手段可以有助于研究者更好的理解和掌握算法的内在逻辑与实际应用。目前对于证据学习这一领域的可视化工具几乎为空白，对于基于证据理论的各类模型算法的运算过程缺少一个展示的工具或者通用框架。

### 存在的问题

由于证据学习理论有别于传统概论统计，致使该领域的初学者对其相关的基础定义存在困惑或者误解，无法理解例如质量函数的含义，使得无法运用这些定义进行模型的搭建，进而阻碍了将基于证据学习的模型算法运用于实践。因目前相关领域发展时间较短，机器学习现有的可视化框架也无法直接迁移于其之上，其主要存在的问题如下：

1. 缺少对基于证据学习的决策树的可视化开发工具
2. 对于证据理论的抽象概念，缺少形象具体的展示方式；
3. 对于证据决策树算法的逻辑理解比较困难；
4. 难以理解证据学习理论知识如何融汇于决策树算法之中
5. 现有算法的可视化多为静态图片，无法体现算法过程的连续性演变过程，增加了理解难度。

## 本文研究内容及目标

本文旨在针对1.2.2中所提到的问题进行研究后，利用动画引擎编写代码生成一个对于基于证据学习的决策树的视频动画，同时提供相关动画的函数API接口。

### 研究内容

本文主要研究如何将证据决策树算法的过程可视化，具体研究内容如下：

1. 研究证据学习相关的基础理论，可视化展示相关核心要点；
2. 证据学习下实现决策树算法，对算法运行过程进行可视化；
3. 搭建Manim动画引擎工具的开发环境进行项目开发

### 研究目标

针对本文的研究内容，制定了以下几项指标：

1. 能够实现一个易于理解、可用于简单机器学习任务的基于证据学习的决策树模型；
2. 使用 Manim动画引擎实现对证据学习的决策树算法构造过程的动画展示；
3. 渲染的动画能将复杂抽象的证据学习理论具体化；
4. 程序具有一定可扩展性接口；

## 本文组织结构

整篇论文分为五章：

第一章介绍了证据理论及可视化的研究背景、研究意义，分析指出了基于证据理论的决策树的可视化发展方面目前的情况与问题，并提出了本文的研究内容和研究目标；

第二章介绍了证据学习理论的基本概念定义、决策树的特点介绍和构造证据决策树的关键原理和步骤；

第三章介绍了基于Manim库的动画开发环境，说明了Manim库的基本结构体系，举例展示了动画制作案例。最后介绍了本文所用的数据集和证据决策树的具体构造方法代码；

第四章主要介绍了证据理论部分基础理论的可视化动画设计，和在此基础上如何设计对证据决策树运行原理的和规则的动画，最后展示和解释了整体视频动画的部分效果图；

第五章总结了全文，叙述本文的主要工作，归纳了创新点，并在此基础上的展望了未来。

# 证据决策树相关理论

本章将介绍证据理论的基础理论、决策树的简单介绍和证据决策树的核心流程以及关键公式的数学表示。

## 证据理论

证据理论，也称信念函数理论或DS理论，作为不确定性推理的框架的一种，与概率、可能性和不精确概率理论等其它框架有着一定的联系。该理论可以结合来自不同来源的证据，得出一定的程度的信念，该信念考虑了所有可用的证据信息。它基于两个思想：从相关问题的主观概率中获取一个问题的置信度（信念）以及Dempster证据融合规则，该规则在基于独立的证据时，可以将各个置信度结合起来[2]。在这种形式中，置信度（也称质量函数）表示为置信（信念）函数，而不是贝叶斯概率分布。概率值被分配给一组可能性，而不是单个事件。置信度本身可能具有或不具有概率的数学性质；它们之间的差异取决于这两个问题的关联程度。换句话说，它是一种表示认知似是而非的方式，但它可以给出与使用概率论得出的答案相矛盾的答案。

证据理论的核心是证据概念，以及如何将不同的证据结合起来，做出推论。限定词“证据”通常是“基于信念函数”的同义词。与贝叶斯概率论一样，它也是一个量化信念的理论。本文考虑的信念函数被假定为具有有限域。在分类任务中，基于证据学习的分类器可以识别出属于没有在训练集中表示的类，因此证据理论在现实中具有更大的实用性。

### 识别框架

定义2.1.1 在常用封闭世界假设下，包含了变量X的所有可能且互斥取值的有限空间称为识别框架 （frame of discernment)，例如：。ω代表变量的真实值，有且只有一个，其为 Θ 中的元素，ω的完全确定状态是当我们能够精确得到ω时，而除了知道ω在Θ中外没有任何其他信息时为完全不确定状态。现实中常见场景是知道变量真实值的部分信息，此时通过定义在 上的质量函数 进行建模[3]。

中所有子集的集合称为 的幂集，用P(表示，其中包含了个元素[5]。例如：若，则P()={∅,1...N,(1,2),(1,3)...(N-1,N),(1,2,3)...}。

### 质量函数

定义2.1.2 质量函数（mass function) 也称为基本概率分配（BPA,basic probability assignment）或基本信任分配函数（BBA,basic belief assignment）[3]。形式上，质量函数m是一个映射，为识别框架的每个假设 分配一个质量值，使得其满足：

某假设A的值m(A)是严格支持假设成立的信念量，同时这种信念量也意味着对假设A的子集的信念分配值我们是完全未知的。

通常，人们只对能够获得 获得 的子集 A感兴趣，称为 的焦元，所有焦元的集合称为质量函数的核，表达了关于X的真实值在集合A中的支持程度，并且仅限于A，并不提供任何关于A的子集的信息。质量函数可以细分为多种类型，分别表达了不同的不确定情况[3]。

本文中提及的质量函数都是基于一个封闭世界的，表明识别框架被认为是可以穷尽的。

### 信任函数

对某个属于的假设A，其确定的总信念量，由其所有的子集的质量函数之和表示，称为信任函数Bel(A)。

函数 是信任函数当前仅当其满足如下条件：

对任意K2和Θ的任意一系列子集，。信任函数Bel可以通过质量函数m获得：

其表达了对命题A的支持/信任程度，提供了信任程度的下限[3]。与贝叶斯概率相比，信念函数理论中，信念不具有可加性，这意味着一个信念和其相反的信念的和可能会小于1，即：

这是由于信念质量可以自由地分配个任何假设，而无需给A的子集。正由于此，信念函数赋予了一个额外的“不确定性维度”。这种额外的不确定性允许信念函数表示未知，这种未知的度量是介于信任函数与下文的似然函数之间。

### 似然度函数

对某个属于的假设A，其不完全确定的总信念量，由所有与假设A产生非空集合的集合的质量函数之和表示，称为似然度函数Pl(A)，或者合理性函数。

函数 是似然度函数当前仅当其满足如下条件：

对任意K>=2和Θ的任意一系列子集，。信任函数Pl可以通过质量函数m获得：

其表达了对命题A的最大不怀疑/可能支持程度，提供了信任函数的上限[3]。似然度函数表达了对假设A的最多的可信程度。似然度函数与信任函数共同组成了未知假设可能性的上下界。

### 众信度函数

对某个属于的假设A，可能可以对其赋予信念的集合的总信念量，由所有属于识别框架且包含假设A的集合的质量函数之和表示，称为众信度函数Q(A)。

设识别框架为，给定质量函数m，众信度函数：。定义如下：

众信度函数反映了可能转移到A的最大信度值[3]。

### 转换函数

为了能够使得决策树做出准确决策，需要将信念函数转化为概率函数，使用Smet提出的公式进行[4]，可将质量函数的可能性转化为常见的贝叶斯概率函数，公式如下：

以上的质量函数，信任函数，似然函数和众信度函数之间存在着一一映射关系，给定一个就可以获得其它函数。

## 决策树

决策树是一种非参数化有监督学习，可以进行数据的分类和回归。通过构建树形数据结构的模型，根据属性值实现目标变量的数值预测。

决策树分类器是一种简单且应用广泛的分类技术。从人类的思想上来说，其就是提出了一系列关于测试特征属性的精心设计的问题。每次收到答案时，都会询问一个后续问题，直到对记录的类标签得出结论。

决策树分类算法通过递归选择最佳属性来分割数据，并展开树的叶节点，直到满足停止循环。最佳分割测试条件的选择是通过比较子节点的杂质来确定的，还取决于使用哪种方法测量测量子节点的信息不一致性。在建立决策树之后，可以执行树修剪步骤来减小决策树的大小。决策树过大会产生过拟合现象，因此，我们需要通过修建初始树的分支的方式，来提高决策树的泛化能力。

简单来说，在决策树算法中，样本由一组特征表示。他们的基本思想是相同的：以数据驱动的方式对样本空间进行分区，并将分区表示为一棵树。

### 构造方式

决策树的构建过程采用自顶向下的贪婪搜索，每个决策节点对应了一个分裂属性，其选择方式根据决策树算法和纯度计算方法不用有所差异[19]。对每个决策节点重复进行“最佳区分属性选择-分割数据集后生成子节点”的过程，直至满足停止条件。最终得到的树就决定了对空间的划分，并且每个子空间有确定的类标或输出预测模型。

构造决策树的主要步骤如下：

1. 将初始数据集全部放入根节点；
2. 从未使用过的属性集中选择具有最强区分能力的属性；
3. 根据所选择的属性，将当前数据集划分，并创建新的节点放入；
4. 当数据集满足停止标准时，当前数据子集所在节点称为叶节点；
5. 重复上述步骤，直到数据集全部划分完毕。

### 常见决策树算法

常见的决策树算法主要有ID3、C4.5和CART等，前两者主要基于信息增益来选择特征，后者则是基于基尼系数。信息增益就是比较在选择某特征划分数据集后，划分的各个数据子集属于同一类的情况表现。但其会有偏向含有值个数多的属性的问题，因而有了使用信息增益了的C4.5。基尼系数也是同信息增益一样的一种衡量数据不确定性的方法，两者运算结果基本接近，但是基尼系数的计算快速，因此也成为了一种主流选择。本质上，三者都是在不断减少数据的不确性，使得数据的纯度不断提升，最后基本属于一类，来达到分类的目的。同时，三者都是建立在贝叶斯传统概论上的，与本文后文所述的基于信念函数的证据决策树不一样。

## 证据决策树

证据决策树是将证据理论引入决策树核心思想，主要目的为提升其面对不确定数据时的分类效果。

与传统决策树不同，证据决策树的各个特征值和标签值全部使用质量函数表示。在此基础之上，重新定义最佳分类属性选择、分类规则和数据区分度这三个主要步骤。由此，生成的叶节点必然会由各个达到该叶节点的数据集的质量函数组成。所以最后，我们将组合节点各个数据的质量函数后作为叶节点的新标签。

### 构造方式

构造证据决策树的主要步骤如下：

1. 获取数据集，各个测试属性和决策属性分别建立识别框架，得到了相应的质量函数，将全部数据放入根节点；
2. 从未使用过的数据集中选择对信念区分性最大的属性；
3. 以该属性的所有幂集元素建立分支子节点，以数据集的该属性的质量函数中的最大值的元素名为该数据的标签，进行分类进入相应的分支子节点；
4. 删除空的和不满足分支节点成立的节点；
5. 当满足停止规则时，当前数据子集所在结点为叶节点，对拥有的数据子集进行合并为一个质量函数作为当前结点的标签；
6. 重复上述步骤，直到数据集划分完成。

### 证据距离的计算

由于各个属性使用质量函数表示，则我们无法使用欧式距离来度量数据之间的差别。所以为评估数据集的差异程度和相似性，我们需要定义一种距离度量方式[5]，对于两个质量函数，计算公式如下：

该公式以识别框架的幂集建立向量空间，各个幂集元素为基向量，以D(A,B)表示各个基向量间的转换方式。

为便于本文之后代码的实现，本文使用Jousselme论文中定义来对上式进行变形[6]，将上述公式转换为如下：

其中为质量函数所在识别框架的幂集中元素。

### 最佳区分属性的选择

我们将根据上述证据距离公式来进行对最佳属性的选择，属性（A）的区分能力越强，属性的相似性与标签属性（D）的相似性越接近。因此，我们设两个数据之间的属性相似性和标签相似性分别为。

由证据距离得来的相似性定义为：

标签属性的相似性与某特征属性的相似性之差的绝对值为二者的相似性偏差程度，则一个属性的区分能力为所有数据集与除自身外各个数据的相似性偏差之和的绝对值所表示，定义如下：

其中D 代表标签属性，为数据集第j个特征属性[5]。

总相似性偏差越小，则代表该特征属性与标签属性越相近，则其拥有的区分能力越强，对分类标签的判断力越强。故选择总相似性偏差最小的属性作为当前数据集的划分属性。

### 停止规则和叶结点处理

在证据决策树中，到达叶节点的各个数据的属性和标签的质量函数维度不一致，无法选择其中之一作为叶节点标签。因此我们需要对到达叶节点的数据集进行处理，使得叶节点由一个统一的质量函数组表示。假设到达该叶节点的数据集数量为n，则使用一个（n+1）的元组表示，其中第一个元素为标签质量函数的均值，其余n个元素为所有各个属性质量函数的均值[5]。设{LN}为到达该子节点的全部数据实例的集合，则定义如下：

其中

此外，我们对叶节点设立了一个边界阈值，以及计算叶节点的属性的平均相似度度量值。当该度量值大于所设阈值时，我们认为当前结点的相似程度接近，其标签的相似度趋于相近，即属于同一种分类结果，此时停止对该叶节点继续分化数据集，认为此时该叶节点数据集划分完成。

### 证据融合规则

为了解决某些问题，我们会遇到需要将代表不同识别框架的质量函数以一种有意义的方式组合起来。通常，每条被我们成为“证据”的信息来源都有一个单独的质量函数表示，然而现在我们需要使用某种组合规则依次融合所有这些质量函数，得到一个代表所有可用证据的信念函数。

在证据决策树中，这个问题就变为了在预测的时候，不同于决策树的待预测数据只会到达一个叶节点，在证据决策树中，待预测数据会到达多个叶节点，所以我们需要对这多个叶节点的质量函数组进行融合，以一个质量函数代表所有“证据”。以多个叶节点的质量函数提供的信息，考虑多个信息来源，可以更好地贴近待预测数据的真实情况，并且可以消除由于部分数据缺失或不准确所带来的错误风险，减少误差率。显然待预测数据与各个叶节点之间的联系度是不一致的，因此在融合过程中，我们要设置一个匹配系数，考虑匹配程度的影响。也可以理解为各个需要融合的叶节点对最终的分类结果有各自的权重值。匹配系数同时也可以代表相应叶节点的证据的可靠性。定义如下匹配系数计算方式，用于定量不同叶节点与待预测数据实例之间的关联情况。匹配系数R计算公式如下[5]：

则结合匹配系数的融合规则如下[5]：

使用该规则可以对待预测数据到达的所有叶节点的质量函数进行融合，得到该叶节点属于标签识别框架的各个幂集元素的可能性。

### 节点的平均相似性

为明确节点何时可以视作代表同一种类别，而成为叶节点停止划分数据集。我们利用证据距离公式可计算相似性的特点，来计算节点的平均相似性值，该值低于某阈值时，我们认为该节点的数据属于同一类，计算公式定义如下：

其中N为该节点的数据集合[5]。

## 本章小结

本章给出了证据理论中识别框架、质量函数、信任函数、似然度函数和转换函数的数学定义，介绍了传统决策树算法的逻辑。引出了证据决策树中证据距离、属性间相似性、叶节点构造和证据融合规则的计算方法以及用处，后文第三章将基于这些公式解释证据决策树的完整构造方法。

# 可视化开发环境及证据决策树算法的实现

本章将介绍本文所使用的动画引擎Manim的基本使用方式，还将介绍证据决策树模型的构造和预测过程，配合实际数据集进行详细计算过程的说明。

## 可视化技术的概念

可视化技术（Visualization）最初起源于计算机科学领域，其目的在于将人脑无法理解的抽象数据、理念和逻辑，以简单易懂的视觉画面呈现，以达到将生涩难懂的知识变得易于记忆和理解。其主要依靠图形界面展示将抽象对象具体化，起初主要在计算机图形学和图像处理技术领域发展。随着科技发展，可视化技术早已不在局限于计算机领域，在医学、教育、人工智能等诸多领域内也发挥着巨大作用。同时，可视化技术也从图形跨度到视频动画，甚至到现在热门的虚拟现实技术，都可以算作是可视化技术的一类。

机器学习领域由于复杂的模型，不便于人们理解和学习其中过程，使得对机器学习的可视化变得非常有价值，常用的绘图库如Matplotlib等对机器学习的知识学习起到了巨大作用。

## Manim库简介

Manim是一个以Python库形式使用的数学动画引擎，由3B1B发行，其原本致力于成为创建解释性数学视频的精确编程动画引擎。该库使用OpenGL作为渲染引擎，拥有基于Python GUI的实时渲染交互式窗口。除了本身自带API外，还可以包含外部包，如Numpy和LaTeX。这为本文将其用于解释证据决策树提供了便利。本文选用该库作为可视化动画的开发工具。为完全使用该库的全部功能，本文的在Liunx环境下配置了相应的系统依赖，主要有如下：FFmpeg、OpenGL、LaTeX和Pango。该库可以基于MacOSX、Windows以及Linux等多操作系统使用。本文中所有代码全部基于Linux下使用Manin库。本文中Manim库在Python3.8版本下进行调用。

### 运行方式

与其他Python库和模块不同，Manim 库运行有一些独有的规则。文件必须含有至少一个Scene(或其子类) 的子类，将编写并需要渲染的场景代码放置于其下construct函数方法内，举例如下（图3‑1）：

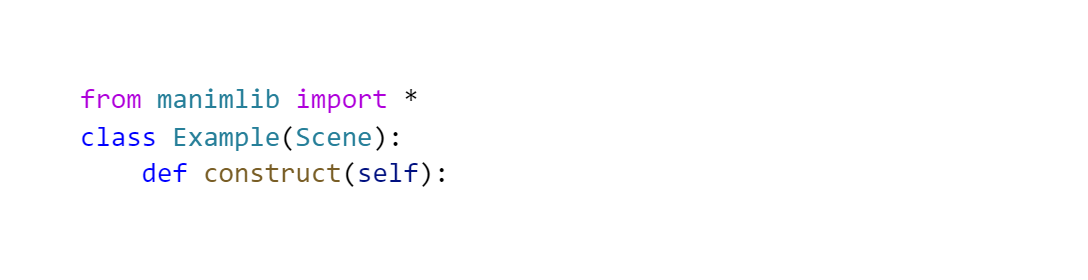
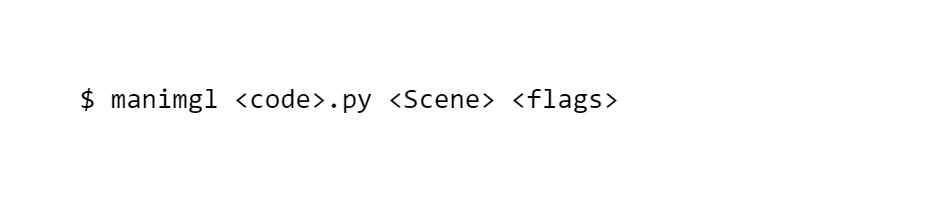


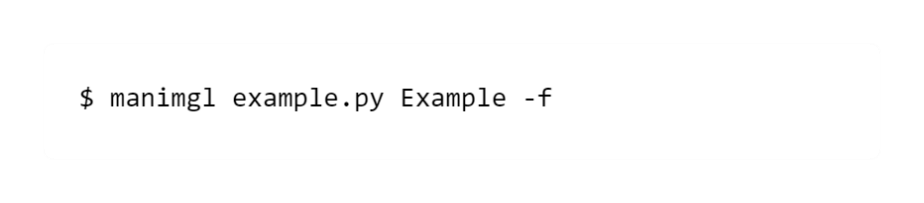
图3‑1 场景例子代码

Manim需要一种特定的格式来运行代码，需要在项目文件所在文件夹打开命令行执行manimgl命令，格式如下：



其中<code>.py 为python 文件名，可以使用绝对路径或相对于当前命令行所在路径的相对路径；<Scene> 为想要渲染的场景，或者一些场景。如果没有写或者写错，且若文件中只有一个 Scene类，会直接渲染这个类，否则会列出所有可渲染类的列表；<flags> 为传入的可选项。

我们举例如下：若要运行example**.**py文件下的Example类，则命令应为：



manimgl为命令符，example.py 为项目主文件名，Example 为需要渲染的画面的类名，-f 为全屏显示的参数，全部参数的作用可使用 --help查看。

渲染运行逻辑图（图3‑2）如下：

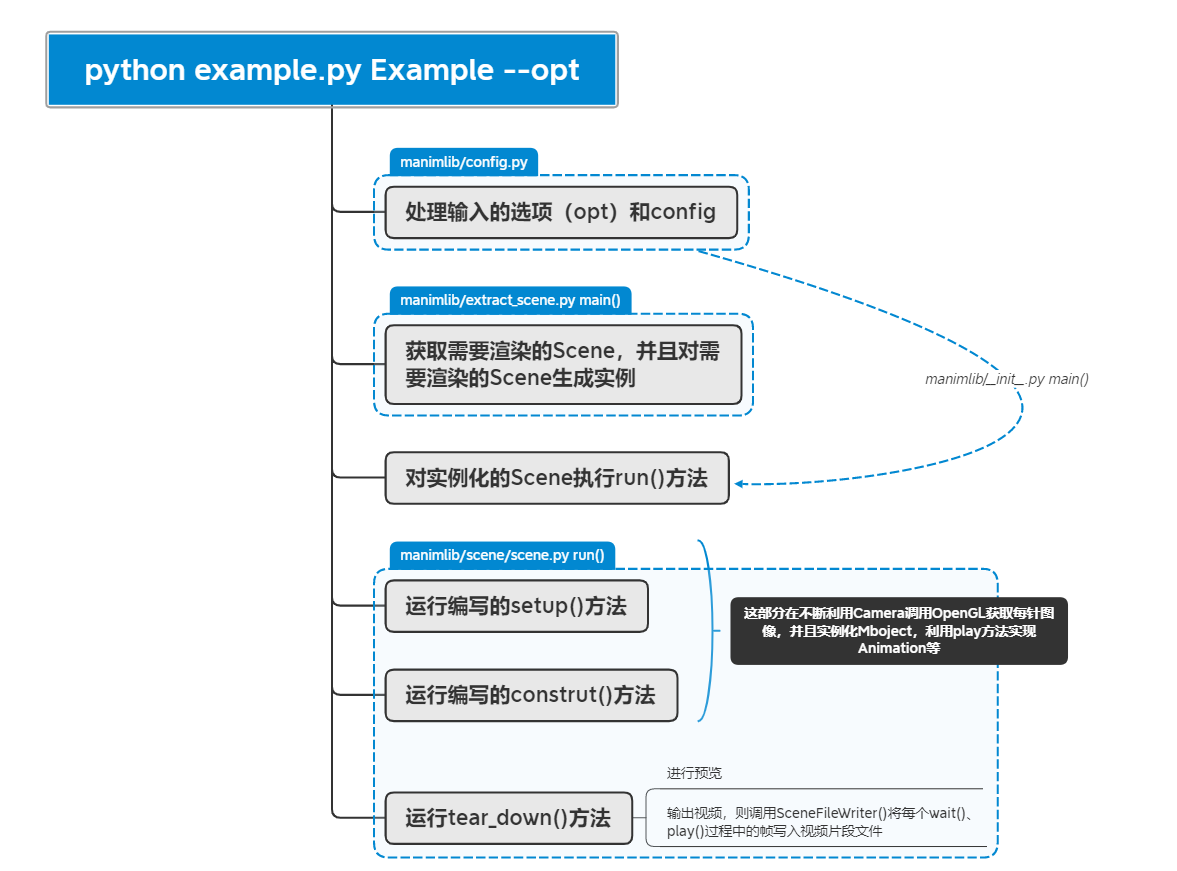


图3‑2 Manim渲染过程

渲染会首先处理命令所传入的处理选项，接着会对需要渲染的Scene子类实例化，然后会运行底层的run函数进行基础画面的渲染。最后才会执行对开发者自行编写的setup和construct函数中的内容。最终使用何种渲染方式是由tear\_down函数决定。

为优化代码调试，减少打开命令行这一重复动作，本文在Python文件中添加system命令，使得可直接使用代码编辑器运行渲染，无需再打开命令行，举例如下（图 3‑3）：

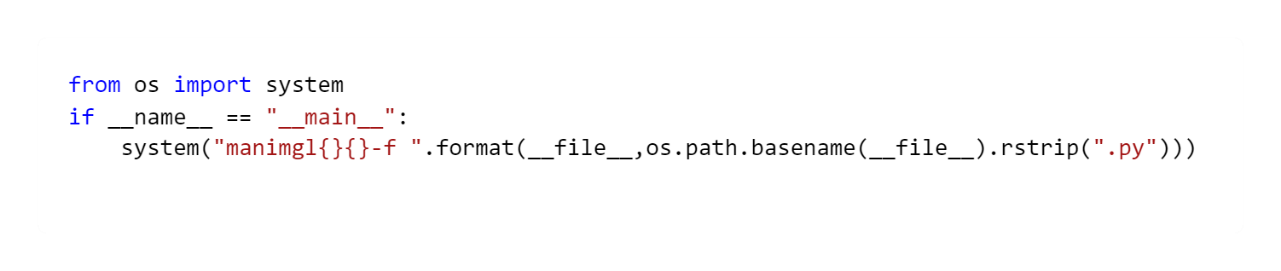


图3‑3 system命令代码

此外，直接运行manimgl命令可以进入交互式编程模式，如下图（图3‑4）所示：

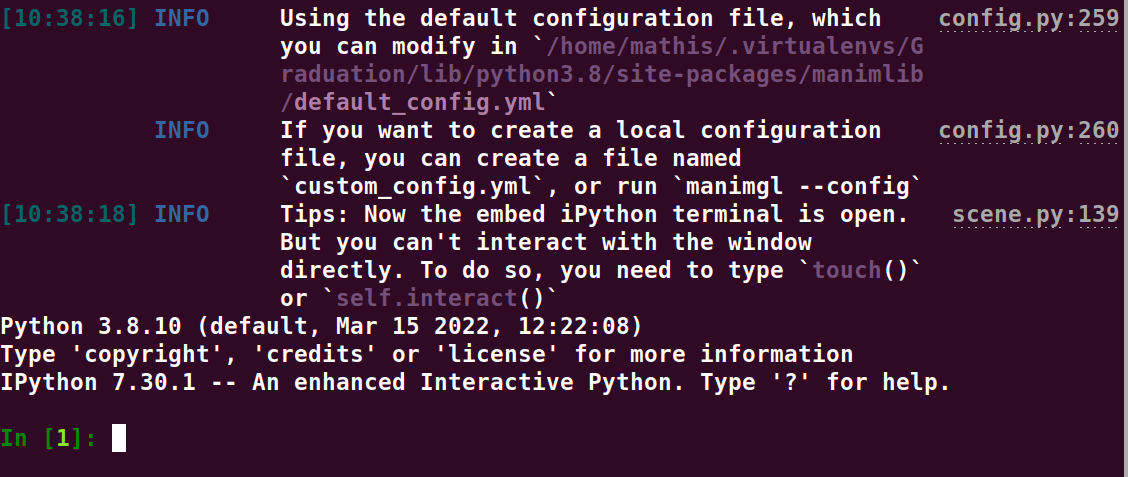


图3‑4 交互式编程页面

### 渲染方式

Manim拥有两套前端渲染模式，分别对应预览和生成常用视频格式。若选择预览功能，在执行manimgl命令后，命令行在显示渲染进度的同时，GUI界面会同时播放动画，此时是使用OpenGL的实时渲染，即是基于Pyglet的，这节省了动画制作的时间，方便调试程序代码。同时，此时可使用键盘与鼠标进行交互；选择执行产生视频文件的时候，则会调用基于FFmpeg的渲染，生成例如Mp4格式的视频文件，逻辑图见（图3‑5）。

除此以外，在文件末尾添加embed函数后，会打开IPython终端，可使用代码进行交互控制。或者继续编写代码，可继续实时生成视频观看代码结果。

所有需要渲染的物件、动作等需要全部在Scene及其子类中传入play函数。

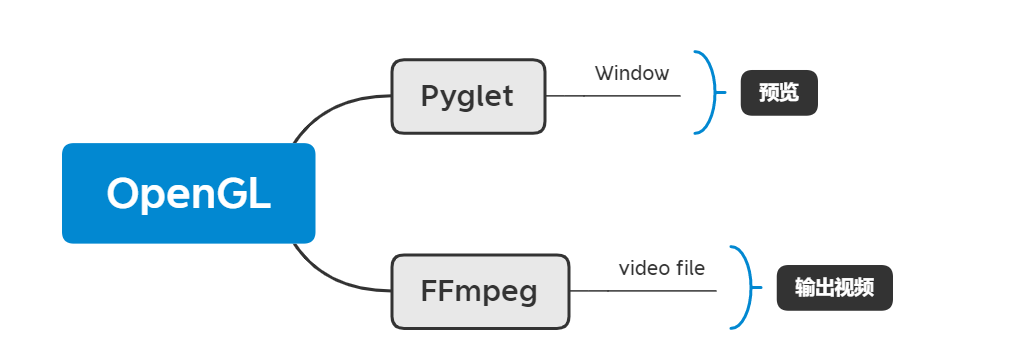


图3‑5 Manim的渲染方式

对于公式的渲染，我们可以通过实例化Manim基础Tex类的方式在动画中添加数学公式。Manim将调用LateX编辑器进行编译渲染。为防止Python对传入的公式字符进行转义，需要向Tex实例传入字符参数时候添加原始字符‘r’。举例如下：

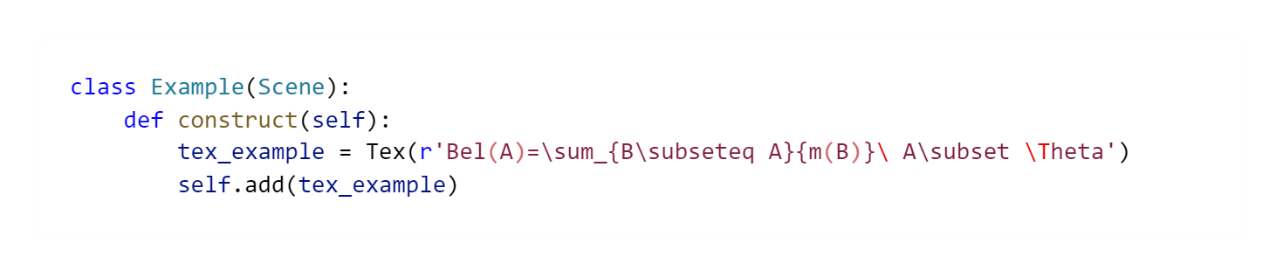


图3‑6 添加Latex公式的代码

我们创建一个第二章中所提到信任函数的公式，传入的参数使用Latex的使用规则画面如下（图 3‑7）：

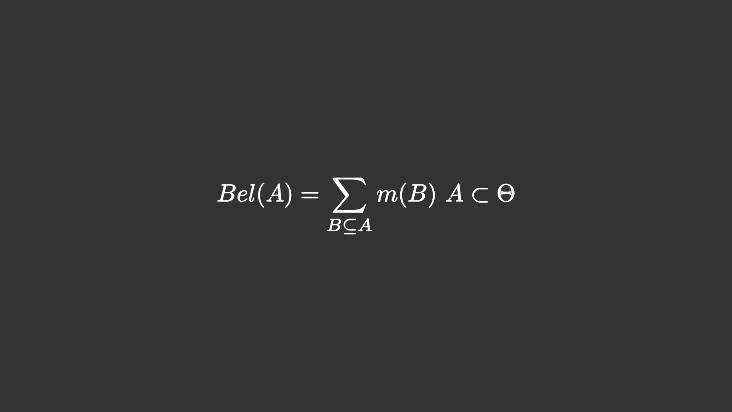


图3‑7 添加公式后的效果图

### Manim 主要目录结构

Manim 库的图形渲染主要依然各种基类的实例化进行控制，所有基础类主要被分为以下几个类别：动画部分、相机部分、基本物件、场景部分和其他辅助类。

基本物件类中，Mobject 类为渲染画面中出现的所有物件的超类，负责所有出现在屏幕上的物件的基础渲染工作，其直接子类有VMobject、PMobjet、ImageMboject和ValueTracker。Mobject类具有的公共属性和方法主要有控制颜色，透明度，反光度，是否固定与画面中坐标等等基本属性。

VMobject类负责使用贝塞尔曲线作为轮廓生成的图形，其创建的实例都是矢量图。PMobjet类是使用点集构成的物体，ImageMboject类为载入外部图片，ValueTracker类为数值类型，不在画面中显示。其类的继承关系可以形象地用图表展示，本文仅选取部分继承关系展示，如下图（图3‑8）：

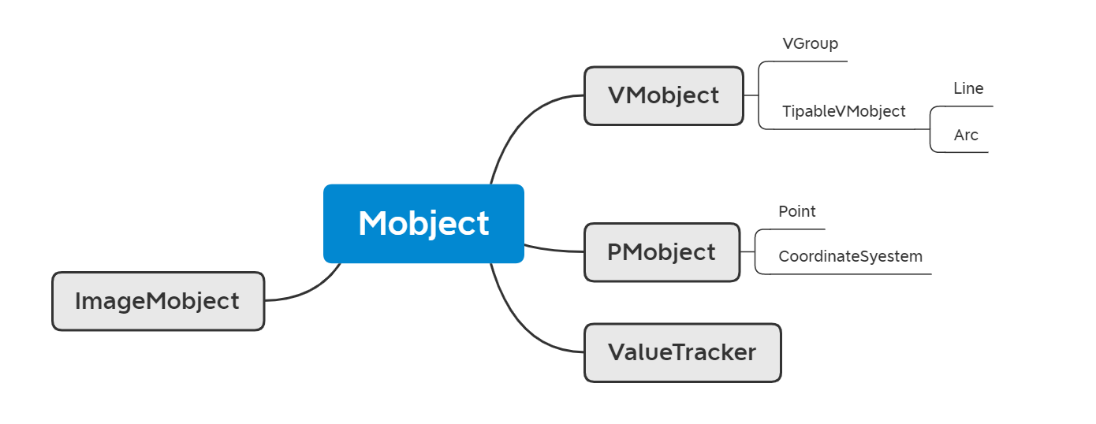


图3‑8 Mobject下类的继承结构

Mobject类还拥有如下的功能模块：

* + - 1. Geometry类定义了常用的几何形体，包括圆、椭圆、矩形；interactive类定义了可供使用的交互组建，比如按钮、滑块。
      2. Coordinate\_systems类定义了坐标轴系统，从中可以绘制多种坐标轴。
      3. Functions类提供了参数曲线类，可以通过参数方程生成曲线的类；Matix类提供了可视化矩阵。
      4. Probability类是与概率论和统计有关的类；svg类为与SVG矢量图载入和文字渲染有关的类；tree\_dimensions类提供三维形体的渲染；vector\_field是与向量场有关的类。
      5. Animation 类为所有动画类的超类，AnimationGroup类可实现多个动画同时渲染的多进程；Fade类负责物件的出现以及消除；Indicate类提供强调画面中某一区域或物体的一系列方式；Movement 类提供移动物件的方法；Transform 类提供物件到物件直接的转换过程的动画，以及文字之间的衔接转换。上述类的继承关系可以形象地用图表展示，本文仅选取部分继承关系展示，如下图（图3‑9）：

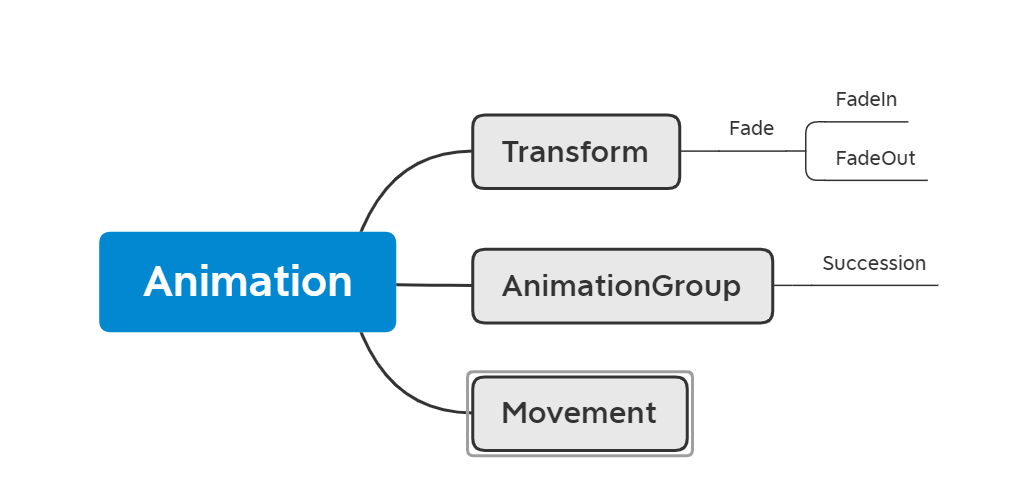


图3‑9 Animation下类的继承结构

* + - 1. Scene 类是最为关键的一个类，所有需要渲染的动画必须以其为超类，并且所有需要渲染的物件必须置于其construct方法下调用。在生成视频过程中，SceneFileWriter方法是必须要用到的，其负责实现视频文件的写入，并在渲染结束后合并为完整视频。
      2. Camera类是一个控制观看视角的类，其提供一些列操作画面视角的功能，可修改视频渲染的每帧图像。

### 基于Manim的可视化案例

我们将编写一个简单的动画过程IntroduceExample程序，来展示Manim制作动画的全流程，程序代码如图（图3‑10）所示。



图3‑10 IntroduceExample程序代码

在上述程序中，我们进行了如下步骤：

1. 导入Manim库；
2. 创建IntroduceExample类，继承了基础场景渲染类Scene;
3. 定义construct方法：
4. 在VGroup类中实例化Circle、Square和Triangle类。三者分别实例化圆形、正方形和三角形。VGroup类可以将多个物件组成整体进行调整，访问其内部物件的的方式类似List；
5. Set\_color 方法可以修改物件的颜色，set\_opacity方法可以修改物件颜色填充的百分比；
6. 使用Tex类传入Latex公式所需的参数，可以进行画面中公式的渲染，公式的编写方法完全与Latex中一致；
7. 因为物件默认生成位置都为原点，我们需要调整VGroup类中各个物件的位置。移动位置的方式有很多，例如move\_to方法、next\_to方法和shift方法等等；
8. 调用AnimationGroup方法，此方法可以使得对物件的动画效果同时进行；
9. ShowCreation方法是一种物件在画面中显示的方法，它将展示物件的整个创建过程，而非之间添加物件到画面中；
10. 所有的动画效果的方法最终都需要传入play方法中，才能最终在画面中生成；
11. ReplacementTransform方法可以使画面中一个图形物件转变成另一个图形物件，并且展示转变过程；
12. Add方法可以将编写的Latex公式添加到画面中，同样其位置也可以进行调整。

我们在文件主函数中预先编写了system命令，因此完成程序的编写后可以直接使用编辑器运行。system命令中将传入我们的文件名和需要渲染的类名，其中-f 代表全屏显示，-w 代表生成高质量的Mp4 文件格式。在渲染完成后（图3‑11），我们将能在项目文件下的video文件夹中找到相应的视频文件。

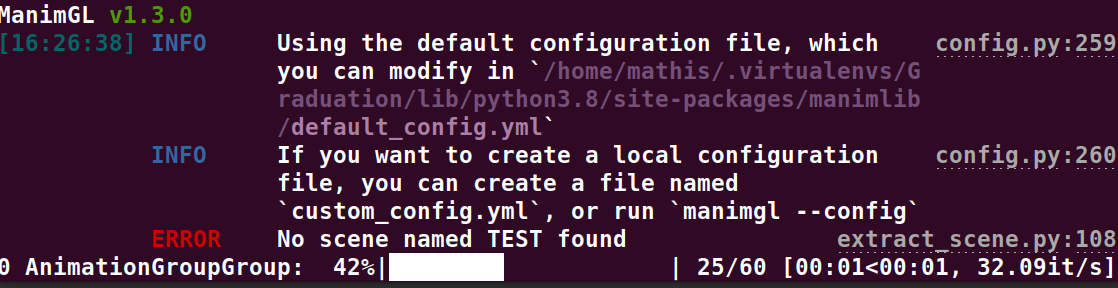


图3‑11 视频渲染中

打开视频保存的文件夹，我们可以发现视频已经保存成功，如下图（图3‑12）所示：

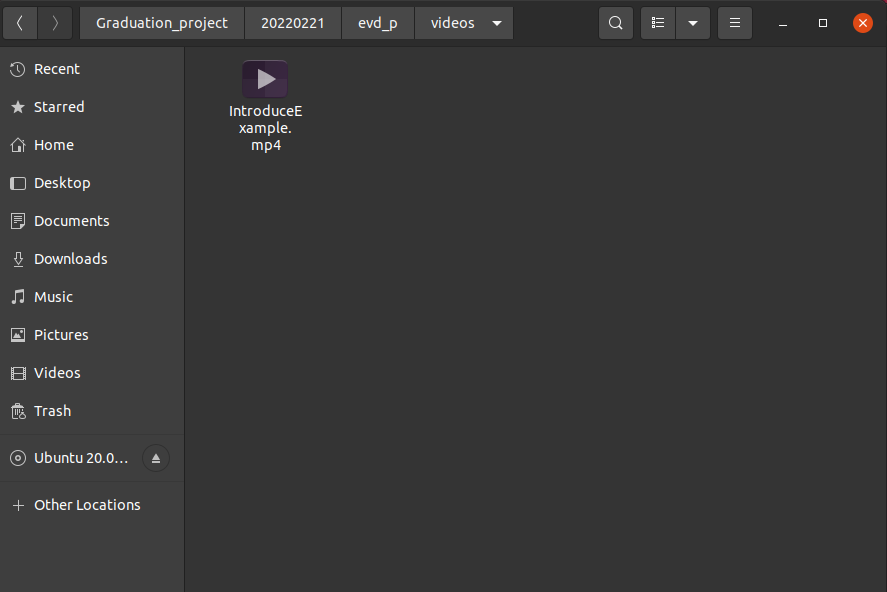


图3‑12 视频保存成功

视频中展示了部分重叠的正方形、三角形和圆形，设置了正方形颜色为蓝色，三角形颜色填充度为50%，图为三者正在创建过程的动画。随后圆形将变换成为正方形，变换完成后将Latex公式添加到画面之中，效果截图如下（图3‑13）（图3‑14）（图3‑15）。

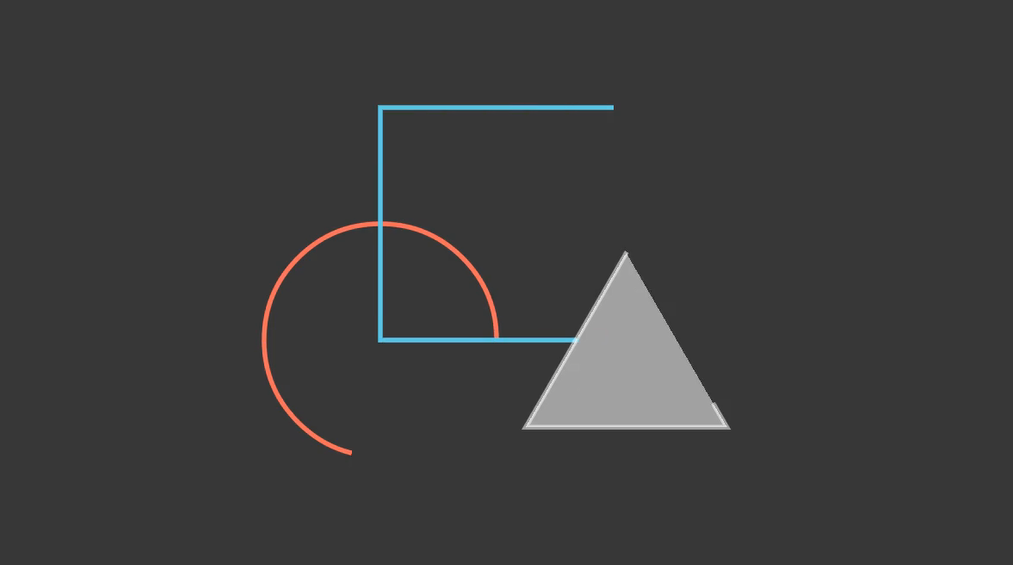


图3‑13 三个物件绘制过程中

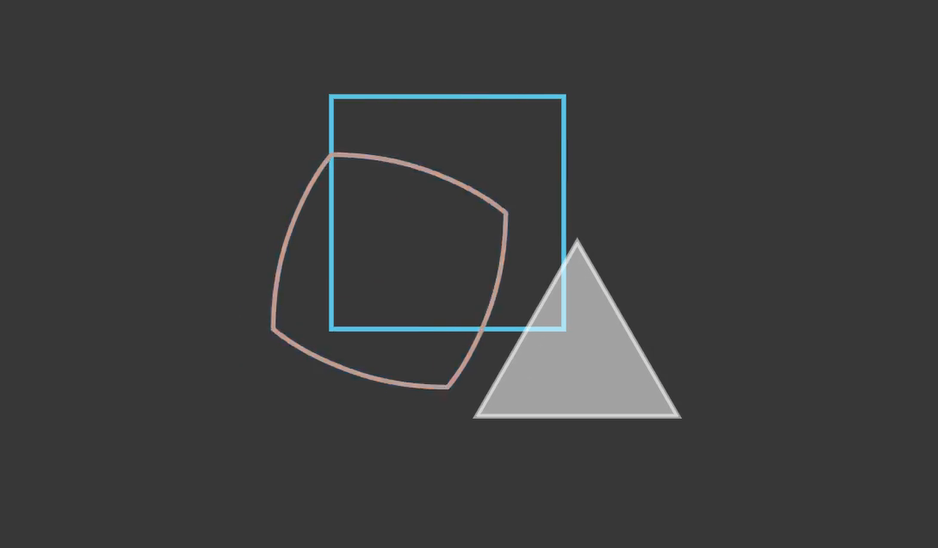


图3‑14 图形变换过程中

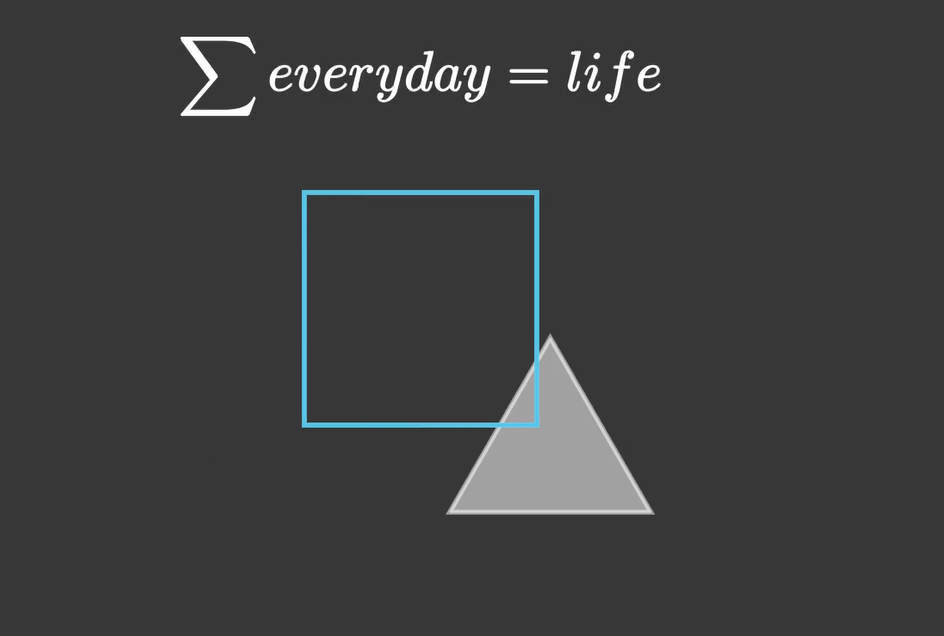


图3‑15添加Latex 公式

## 证据决策树模型的搭建

我们设代表数据集的各个属性值，表示数据集中的各个实例，代表各个属性所建立的识别框架。一个数据集的某个实例()在某属性()上的质量函数定义为。为数据所属标签的识别框架，即最终分类的可能结果。

### 质量函数的实现

本文的质量函数类是在Python字典类之上实现的，支持归一化和非归一化的质量函数。我们假定基本的识别框架是离散不相交的，可以使用标准的字典方式添加、更改、删除假设的及相关的质量函数的值。每个假设都是可以是一个任意序列，该序列会自动转换为frozenset类型，表明它的元素必须是可散列的。在创建质量函数时候，可以选择质量函数、信任函数、似然度函数和众信度函数其中任意一种，因为四者之间可以相互转换，且具有唯一性。在本文后续中，默认选用质量函数这一构造方式。

### 数据集定义方式

我们选取EDTs提供的数据集作为本文构造证据决策树的数据来源。其具有三个特征属性，一个标签属性。分别以这四个属性建立识别框架，生成相应的质量函数，以双重列表嵌套质量函数的方式存储。本文模型是基于此数据集实现一个三分类分类器。

在建立各个属性的质量函数时，我们选择属性的离散不相交元素为基元素，以此构造识别框架，则该识别框架的幂集所有元素为质量函数的全部可能取值的集合。我们将以属性ability作为例子进行说明。我们选取{good,ordinary,bad}三个离散且不相交的元素作为ability的识别框架的基元素，为了便于说明，我们以G、Y、B分别代替good、ordinary和bad，简记为{G,Y,B}，则，识别框架。

按照第二章中定义（§2.1.1），其幂集为。则数据集中的数据表示为（以第一个数据的第一个属性为例）：。

本文所使用的全部数据集如下（图3‑16）：

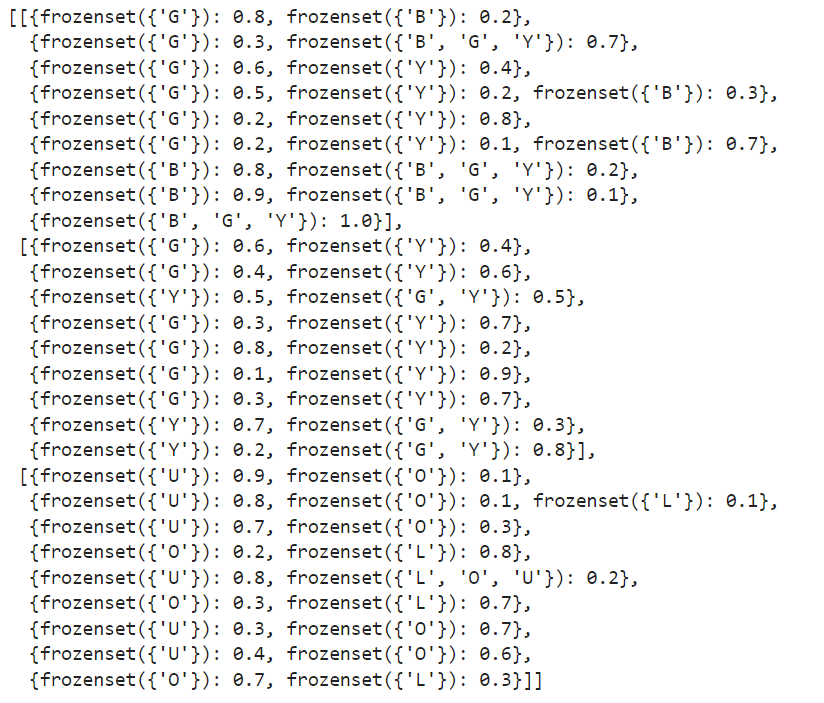


图3‑16 数据集存储结构

### 证据决策树的基本功能函数

我们使用递归和多叉树的方式构造证据决策树。在节点中，若是非叶节点，则存储数据子集、当前节点的子节点和当前节点的最佳分类属性；若是子节点，则存储到达该节点的数据集。由于在算法过程中需要产生多叉树，因此使用字典存储各个节点的子节点地址和节点名称，使用该结点的属性所建立的识别框架的幂集元素作为字典的关键字。

根据第二章证据决策树（§2.3）中所述编写代码（图3‑17图3‑18），在这里我们简单介绍关键代码的功能：

1. 根据定义distance\_dbba函数计算任意两个数据的质量函数之间的距离度量值；
2. 根据定义map\_dis函数计算数据集某个属性基于标签属性的相似性；
3. 定义get\_best\_feature函数得到当前数据集下的有最佳区分性的属性
4. 定义stop\_criteria函数为递归函数的终止条件；
5. 根据定义tru\_lev函数对叶节点的平均相似度进行评估；
6. 根据定义similarity\_attribute函数对数据集属性的相似性进行计算；
7. 定义set\_operation、combine\_two\_dict、combine\_two\_mf等函数执行集合交、并等相关集合类操作；
8. 根据定义relation\_of\_two\_instance函数计算预测数据与各个叶节点之间的匹配系数，也可认为是各个叶节点对最后分类的权重值大小；
9. traver函数负责对传入数据进行预测；
10. create\_tree函数为构建证据决策树的递归函数；
11. 根据定义structure\_of\_leaves函数对叶节点数据集处理后得到叶节点统一标签；
12. 根据定义 combine\_rule函数对多个叶节点数据进行融合。

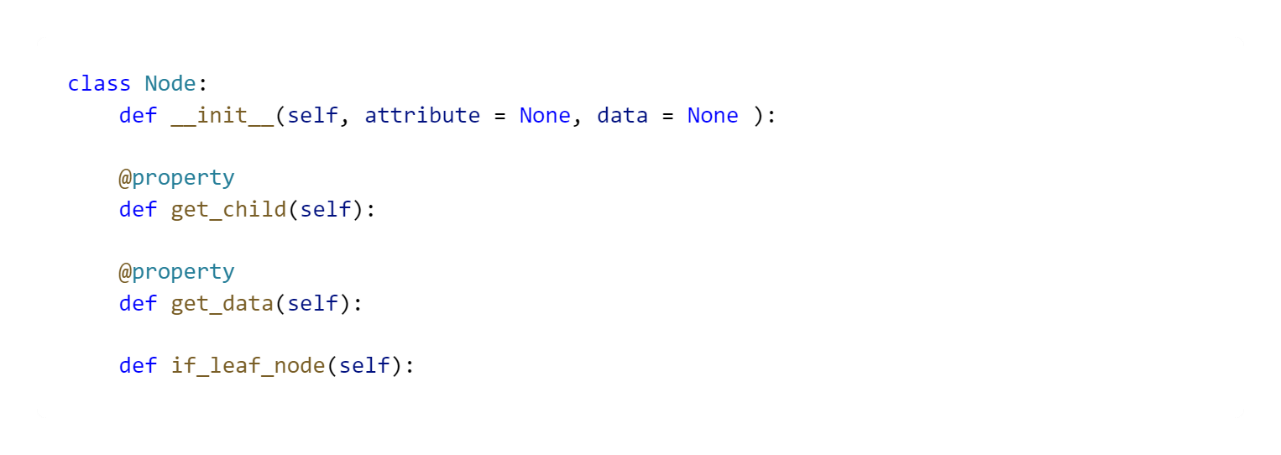


图3‑17 Node类



图3‑18 EvidenceTree类

### 证据决策树的构造逻辑

我们将使用§3.3.2中定义的数据集来展示证据决策树的构造过程，我们假设数据拥有9个实例，3个特征属性，分别以、、建立识别框架，注意各个识别框架中的元素都独立不相关（仅字母可能相同）。此外，还有1个标签属性，以{}建立识别框架。具体构造过程如下：

1. 根据数据集全部实例，首先使用map\_dis计算比较3个测试属性基于标签属性的差异性。我们得到，和。由差异性数值越小，该属性的区分能力越强，我们选择Ability作为当前最佳属性。明显没有任何停止分类的判断符合，因此Ability放入树的根节点作为区分属性。其识别框架为，推出幂集为则ability生成8个子节点，分别由幂集各个元素作为子节点名称。此时，对数据各个实例，选取质量函数最大值，若该值大于（=0.5，实验所得），认为其有效。则进入该质量函数对应的幂集元素相同的子节点，例如实例，最大值为0.8 （>0.5），对应幂集元素为B，则该实例进入以为名称的子节点。若该值不大于α，则将当前值与次最大值相加，重复直到累加值大于α。此时实例进入累加的质量函数相应的元素的并集集合相对应的子节点。同理处理其它实例，完成后删除空的子节点，我们得到如下四个子节点和其拥有的数据子集：，，和 。
2. 对于子节点，不符合停止规则，再次以现有数据集计算差异性，可得和。在上一节点使用过的ability属性不再使用，因为无论数据如何变化，属性的区分能力不受影响。则此时最佳分类属性为property。再次重复第一步的操作过程，可获得两个新的子节点，此时利用节点平均相似度计算公式，得到该叶节点的平均相似度（0.88）大于阈值（=0.8，实验所得），认为此时该节点的实例基本属于同一类别，此时该节点为叶节点，停止继续划分数据集。利用叶节点处理公式中规则计算均值作为叶节点的标签。对所有子节点递归重复操作，我们可以得到一颗完整的证据决策树。

### 证据决策树的预测

我们假设需要预测的数据实例为，和。

预测时，根节点的属性是全数据集区分能力最强的属性，由此往下的子节点属性区分能力初步下降。因此，在根节点数据需要传入该实例的当前属性的质量函数拥有的所有的元素相应的子节点，对而言，即是传入G和Y子节点。除根节点以外的节点，数据实例传入当前属性的质量函数最大值所对应的元素的子节点。迭代这个过程，直到数据实例达到叶节点。由此，我们很容易发现，在证据决策树之中，预测数据实例会达到多个叶节点，而非一个明确的叶节点，这与传统决策树是不一致的。这样就自然想到，我们需要将预测数据实例到达的多个节点进行融合处理。叶节点各自拥有不同的质量函数组表示，其对预测数据的影响是不同的，故我们在融合各个叶节点时候赋予权重系数。我们使用第二章中所定义的匹配系数来作为融合时候的各个权重。匹配系数大小与测试数据与叶节点数据的相似性决定，两者之间差异性越小，则匹配系数越大，对预测数据最终的影响比重越高。则对于，其达到的叶节点为 和 ，与这两个叶节点的匹配系数分别为0.87和0.51，说明其与叶节点的相似性更高，我们使用融合规则可得：。

此时得出的概率为以分类种类标签建立识别框架的幂集的结果，表明了该预测实例属于这各个幂集元素的可能性大小，其具有一定范围内的重叠性。但是实际上，我们在多数情况下，想知道该预测数据实例属于的是哪一种，以及其传统概率大小，这更符合人们对事物的认知。故我们可以使用转换函数，将其从信念函数率转换到概率函数。可得：。

如此，我们可以得到预测结果属于哪一类的概率函数，更符合人们对事物判断的日常习惯。同时为我们提供了与传统机器学习算法比较的途径。

## 本章小结

本章介绍了可视化技术的背景，介绍了本文所使用Manim数学动画引擎的基本开发方式，展示了一个实际开发程序代码。随后介绍了证据决策树算法代码中质量函数类的定义，数据集的存储方式和构造证据决策树的核心代码解释。

# 证据决策树可视化具体实现

本章将介绍如何将证据学习基本概念和证据决策树关键函数定义的可视化，以及证据决策树的可视化动画的制作思路和过程。最后将简单展示两个原创API接口的使用。

## 主要核心概念的可视化

我们将使用第三章中实现的证据决策树算法，基于Manim为其添加动画。为保持证据决策树算法原有功能的独立性，我们使用原型模式，克隆证据决策树类，并且将其置于Manim框架下。为确保动画的渲染，我们将按照Manim库规则，将证据决策树动画类继承Object类后在Scene类或其子类中创建进行渲染。

### 识别框架的动画

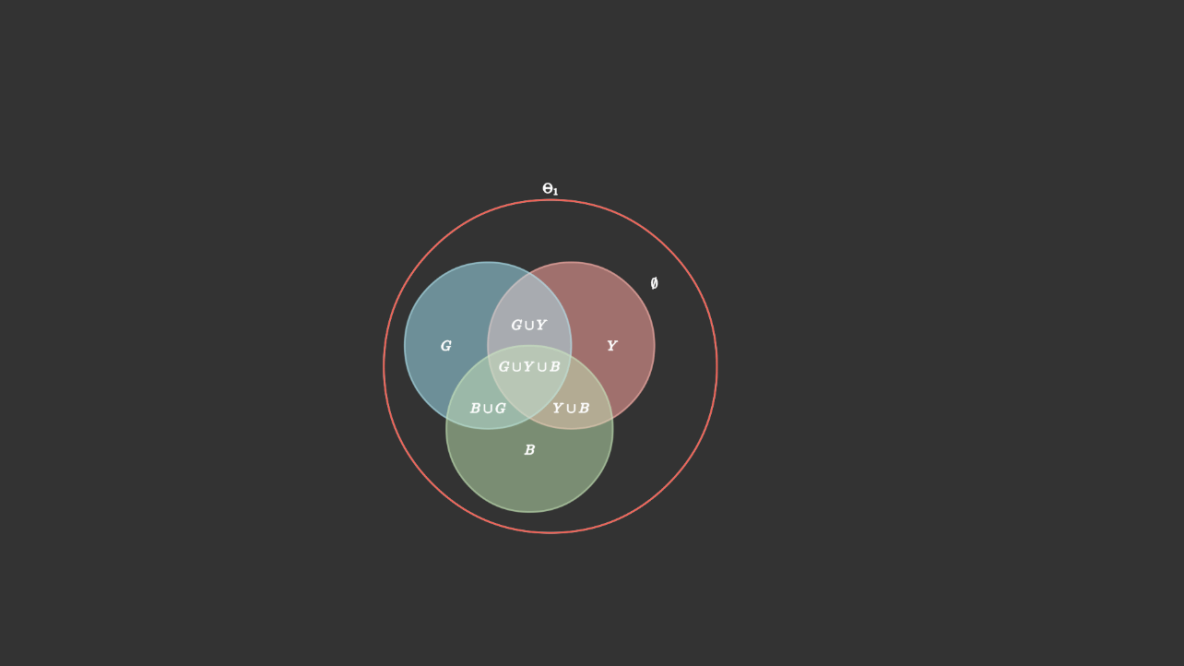


图4‑1识别框架的可视化图

我们以集合圆的形式表示识别框架的基元素，集合圆的相交过程代表着幂集的生成，即组成了离散不相交的识别框架，如图所示（图4‑1）。注意此时G元素所代表的不再是整个圆，而是圆去掉重叠部分，同理B与Y元素。

### 信任函数与似然函数的动画

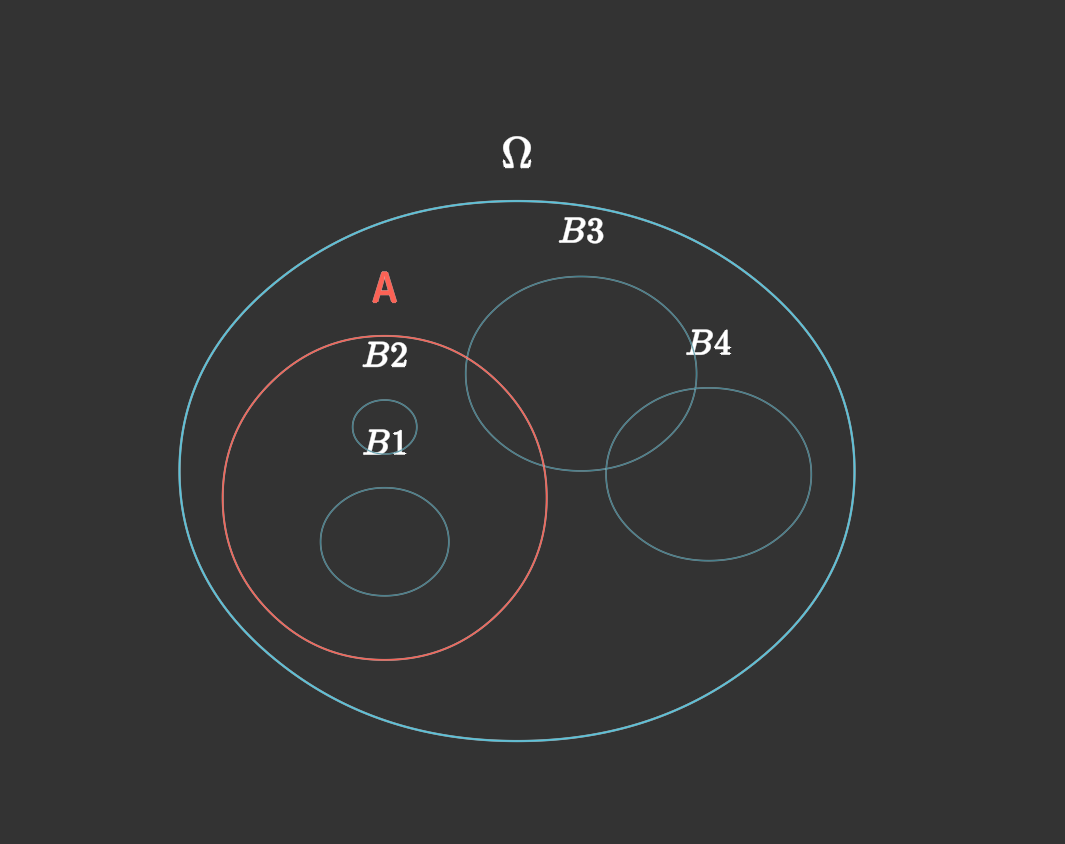


图4‑2信任函数和似然度函数

假设我们有如图的识别框架，当我们需要对的集合A进行质量函数进行评估是，可以看到全部位于A中有B1和B2，与A有交集的有B1,B2和B3，根据信任函数和似然函数定义（§2.1.3、§2.1.4）可知， 如图（图4‑2）可以看出信任函数表明可以一定可以推测出A集合的可能性，而似然表示可以推测出A的最大可能性。除次之外，我们将上述转换到映射到一维线性上，可以得到如图（图4‑3）：

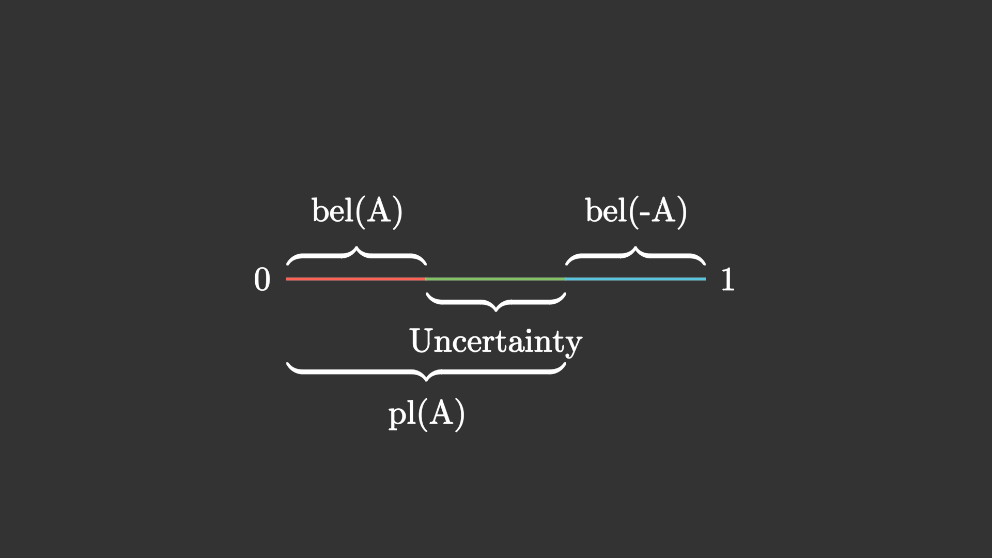


图4‑3 bel与pl 的理解图

从此动画中，我们展示了信任函数和似然函数分别表示数据源的未知性的上下限，将传统的数据具有一个准确值，变为一个区域值。

### 树结构的动画

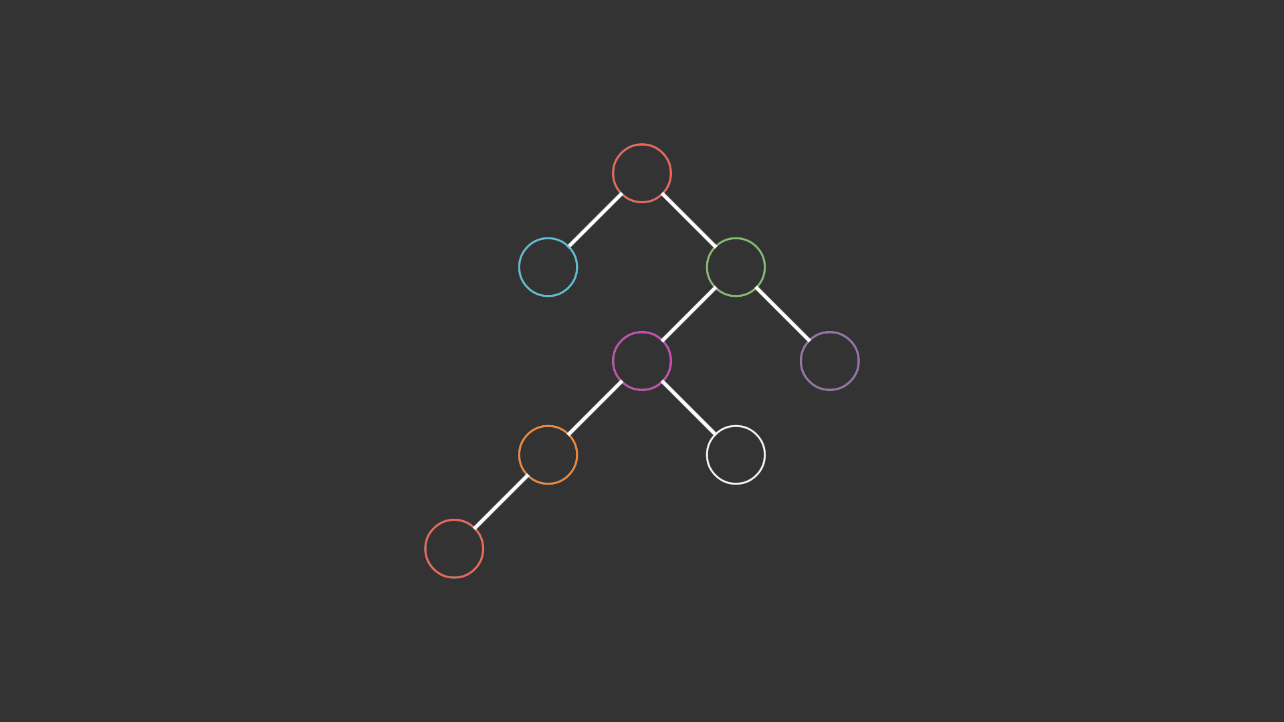


图4‑4 树结构效果图

我们编写ManimTree类实现树形图的构造（图4‑5）生成效果图（图4‑4），ManimNode类负责节点的构造。ManimNode类部分函数功能如下：

1. show方法实现传递需要渲染的物体到Scene类；
2. get\_form方法获得该节点所需要外部形状；
3. ret\_location方法获取该节点的在画面中的位置；
4. is\_leaf\_node方法判断该节点是否为叶节点，如是则不需要渲染节点间的连接线。

为使得之后可以向各节点和树枝添加文字，在初始化ManimNode类时，使用Manim的VGroup类将外部形状类与文本框类进行组合，组成联合体，进行统一控制。

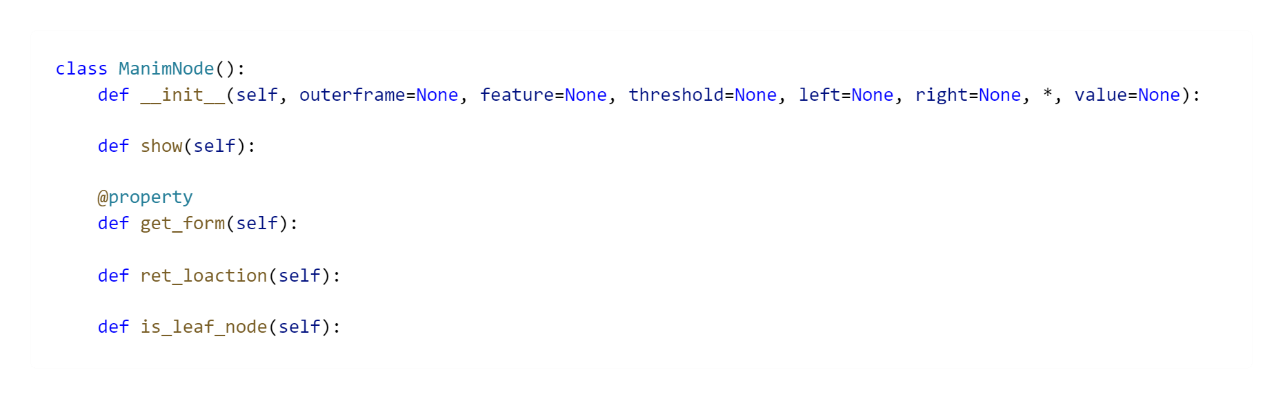


图4‑5 ManimNode 类

ManimTree类（图4‑6）部分函数功能如下：

1. add\_left/right\_node、 add\_l/r\_line等函数负责为树添加节点和节点间连接线的所需基本参数，如：起始点坐标，颜色，大小，阴影偏差等相关属性；
2. grow\_line\_tree、grow\_tree方法实现树结构动画创建过程中的动画效果的设置；
3. traverse\_tree\_manim实现对预测数据预测过程中数学变化已经所经历节点的记录，为之后渲染移动效果提供节点路径。



图4‑6 ManimTree类

### 机器学习中的决策树的动画

在将由传统机器学习决策树训练好的模型传入绘制树结构的类中后，绘制类将会遍历所有节点，记录叶节点的数据子集信息和叶节点的标签类，记录非叶节点的划分属性名称和边界值。同时生成新节点，添加外围形状信息，同时自动进行节点在画面中位置的排序。完成树的遍历后，在调用渲染时候，将分步渲染（图4‑7、图4‑8）。

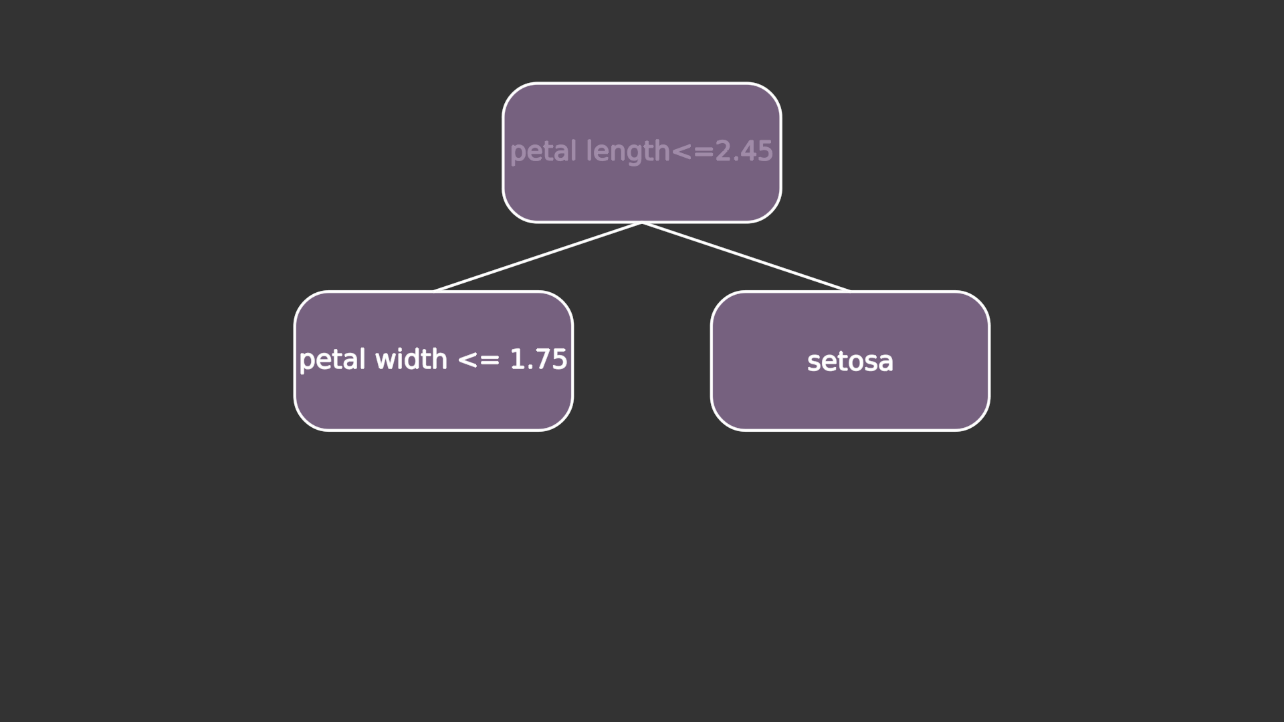


图4‑7决策树生成过程图

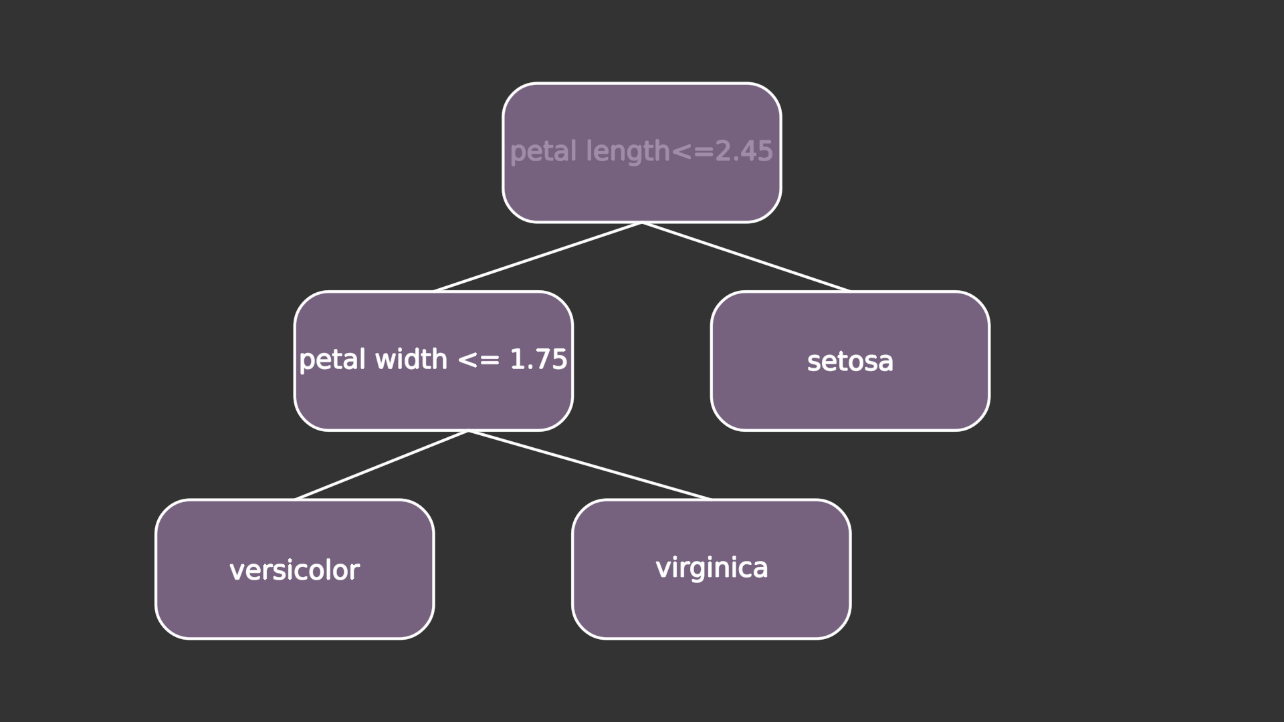


图4‑8决策树完整生成图

### 证据决策树的动画

同上段所述基本一致，在将证据决策树模型传入后，动画将生成从一个根节点开始生长到整个证据决策树，如下图所示（图4‑9）：

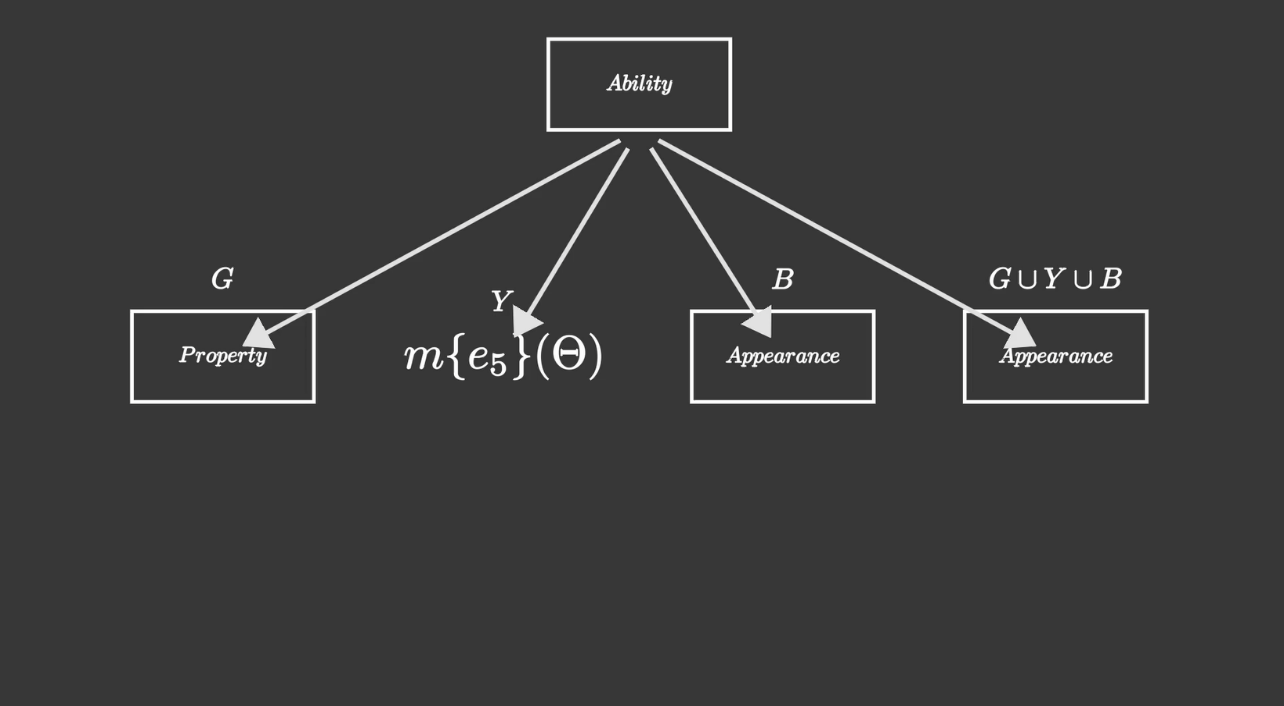


图4‑9证据决策树生成过程图

### 叶节点处理的动画

我们采用实际数据进行运算的方式，介绍如何将叶节点下多个数据合成一个的过程。使用数据集中e13和测试数据e0为例，如下图所示（图4‑10）：

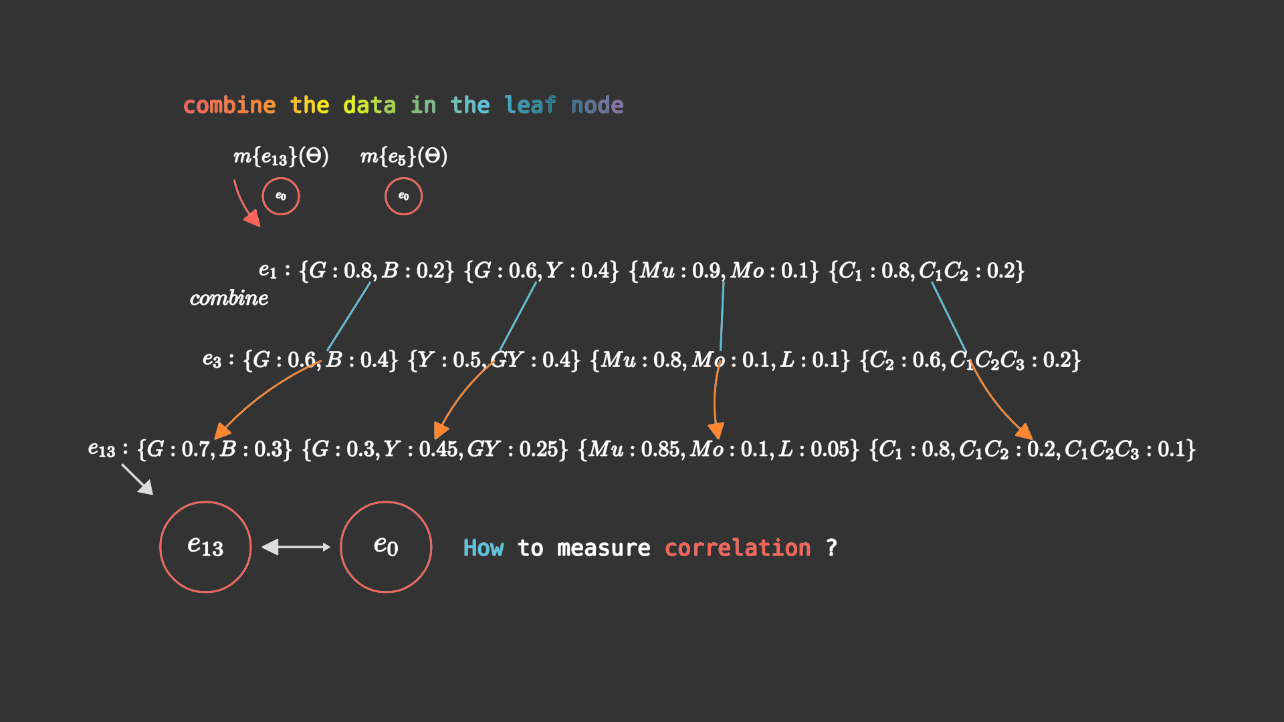


图4‑10叶节点数据集的质量函数融合解释图

## 动画整体设计逻辑

动画将先介绍树结构，随后结合鹫尾花数据集实现机器学习中决策树算法的实现。之后将介绍证据理论的基本概念，由此引出基于证据学习的决策树的介绍。在最后一部分中，我们先介绍证据决策树的整体算法思路。随后将使用第三章中定义的数据集进行一次实际的证据决策树得构造及预测过程的逐步展示。所以，本节动画将分为三个部分。

树结构与传统决策树部分：展示树数据结构的基本形式。本部分将讲述树结构的各个组成部分，以及决策树的分支属性的选择方式、分支停止规则、递归方式等。最后在坐标系中展示鹫尾花数据集通过决策树算法后的划分过程。

证据理论概念部分：将举出实际例子来展示证据理论概念的实际运用和计算方式，从最简单基本运算起，为之后证据决策树中相关复杂计算做铺垫。

基于证据学习的决策树部分： 阐述如何在证据学习框架下实现决策树模型。从数据集定义的不同，到新的数据集间差异性的计算判断方式，以及不同的树的构造方式和叶节点再处理。结合实际的数据集，展示所有每步的计算结果、数学变化和树结构的每次变化。

### 动画渲染过程

本节按照上述动画设计方案进行动画渲染，仅对第一部分的动画渲染过程进行说明，其余部分代码与之类似，不再赘述。完整代码见附件源代码。三部分动画渲染文件代码分别为frame1.py、frame2.py、frame3.py。

树结构与传统决策树的动画渲染，代码见图（图4‑11）：

1）定义树结构场景的基类TreeScene :

1. 该类直接继承Scene类；
2. 定义初始函数setup;
3. 定义添加树结构的函数add\_tree。在本方法中，调用前文提到的ManimTree类的构造函数和Tree\_text类的构造函数，Tree\_text类直接继承VGroup类，负责文字和外形组合功能。
4. 定义展示累的节点的函数show\_tree\_node，该函数将遍历渲染所有节点；
5. 定义连接节点的线段函数show\_tree\_line，该函数将按照子节点地址连接节点；
6. 定义绘制动画效果的show\_plot函数，负责渲染时候的动画效果。

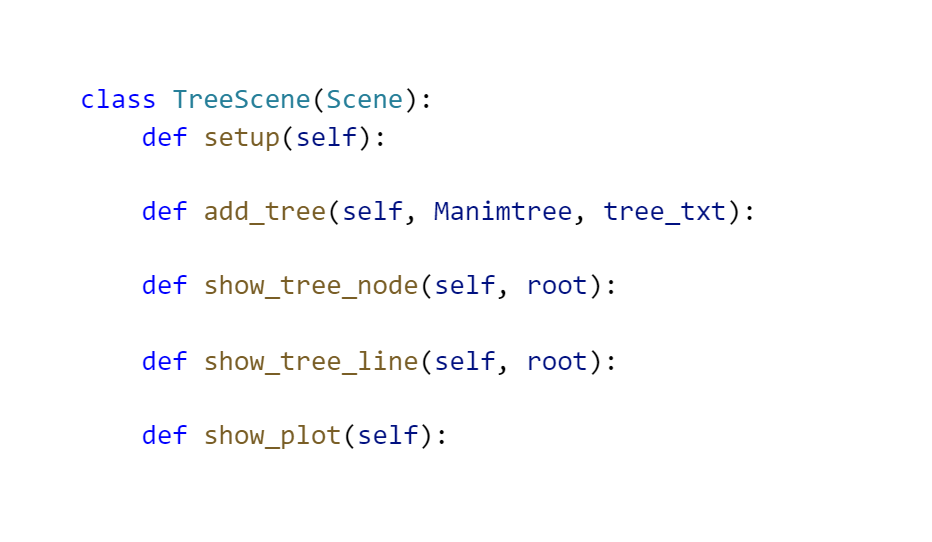


图4‑11 TreeScene 类

2）定义传统决策树的场景类TreeShowScene类，代码见图（图4‑12）：

1. 该类直接继承（1）中定义的神经网络场景TreeScene类；
2. 定义渲染过程construct函数，所有的动画效果置于次函数内；
3. 定义split\_iris\_data函数，负责数据集的加载导入和划分；
4. 定义训练函数traintrain\_data 进行训练；
5. 定义change\_data\_to\_dotcloud 函数将数据转换到云点图。

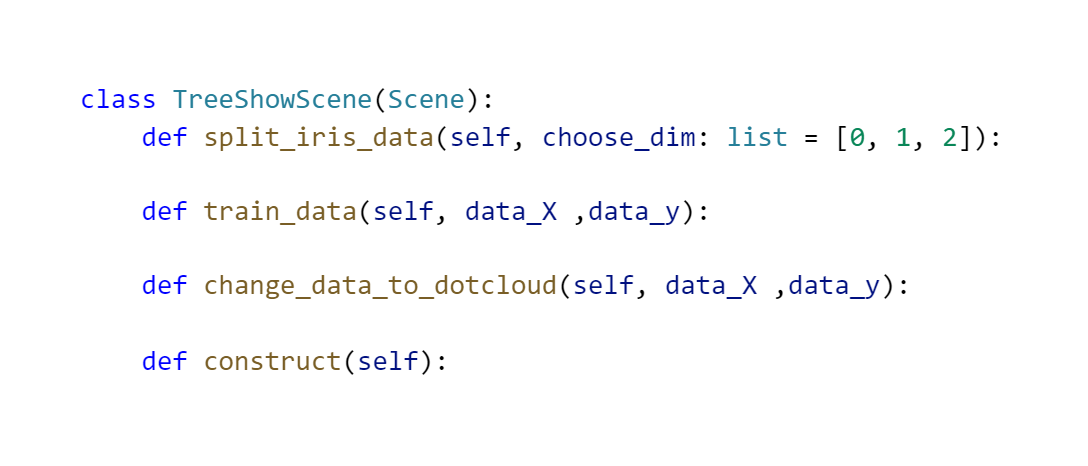


图4‑12 TreeShowScene 类

3）定义坐标轴类CoordS类：

1. 该类继承ThreeDAxes类
2. 定义add\_dotcloud函数在传入处理好的数据集后，将生成相应的点云图；
3. 定义set\_frame函数负责旋转镜头视角；
4. 定义set\_surface函数实现添加分割数据集的动画效果；
5. 定义set\_colro 函数负责对分类进行上色区分。

最终所有类在frame1.py中实例化生成渲染，我们可以得到各类的继承关系如下图（图4‑13）：

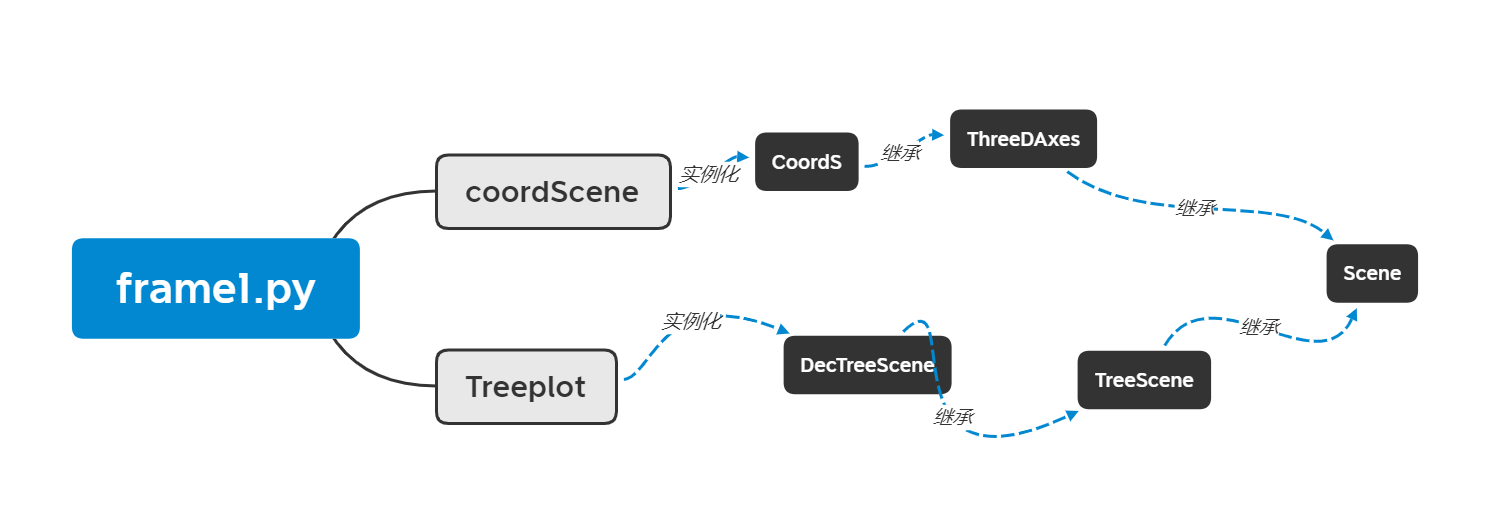


图4‑13第一部分动画的类继承关系

## 整体效果展示

本节介绍了完整的项目框架，及部分动画运行结果的展示。

### 程序文件框架

本文项目文件夹没有建立在Manim库的指定文件夹下，因此需要将Manim库路径添加到编译器默认路径中。在本项目文件中（图4‑14），主要文件有：\_\_pyache\_\_为调用模块的已编译文件、数据集导入文件data\_loader.py、树结构框架文件tree\_manim.py、证据决策树算法文件Edts.py、质量函数定义文件pyds.py、树结构与传统决策树的动画frame1.py、证据理论概念的动画frame2.py和基于证据学习的决策树的动画frame3.py

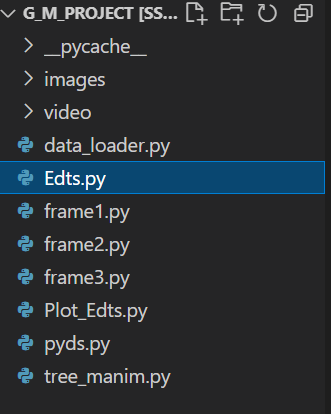


图4‑14项目文件目录

### 程序使用说明及代码运行结果展示

通过使用（§3.2.1）中所提到的运行方式进行渲染后，我们将得到完整的视频动画，下面将截取第三部分证据决策树动画中部分图片进行展示。

1.证据理论基础概念的运算举例（图4‑15）

在视频动画中，为了能让观看者对证据理论中信念量量的计算有一个大致感受，我们举了一个简单的计算例子。我们假设如下情形：一个测量车辆颜色的传感器在某次检测后，得到车是黑色的信念量为0.4，车是棕色的信念量为0.3，完全不知道的信念量为0.1。由第二章定义可知，总的信念量为1，则推导出车是黑色或者棕色的信念量为0.2。如此，我们通过第二章中定义的信任函数和似然度函数的计算方法，可以得到如图中所示的计算结果。我们观察可以发现信任函数是车为黑色的这一假设的最低可能性，似然度函数为车是黑色的这一假设的最大可能性。它们之间的差就可以代表车是黑色这一假设的未知程度，通过这样的一种表达方式，我们就可以为该假设添加一个不确定区间，在这个区间内我们对假设完全无知。我们可用{“黑”：0.4，“棕黑”：0.2}记录车是黑色这一假设，而非车黑色的概率为0.5。这样的定义方式区别于贝叶斯概率的定义方式，贝叶斯概率会给定假设一个准确值，也就是假设只有两面性，无法体现现实世界中，通常假设可能具有其他未知的情况。

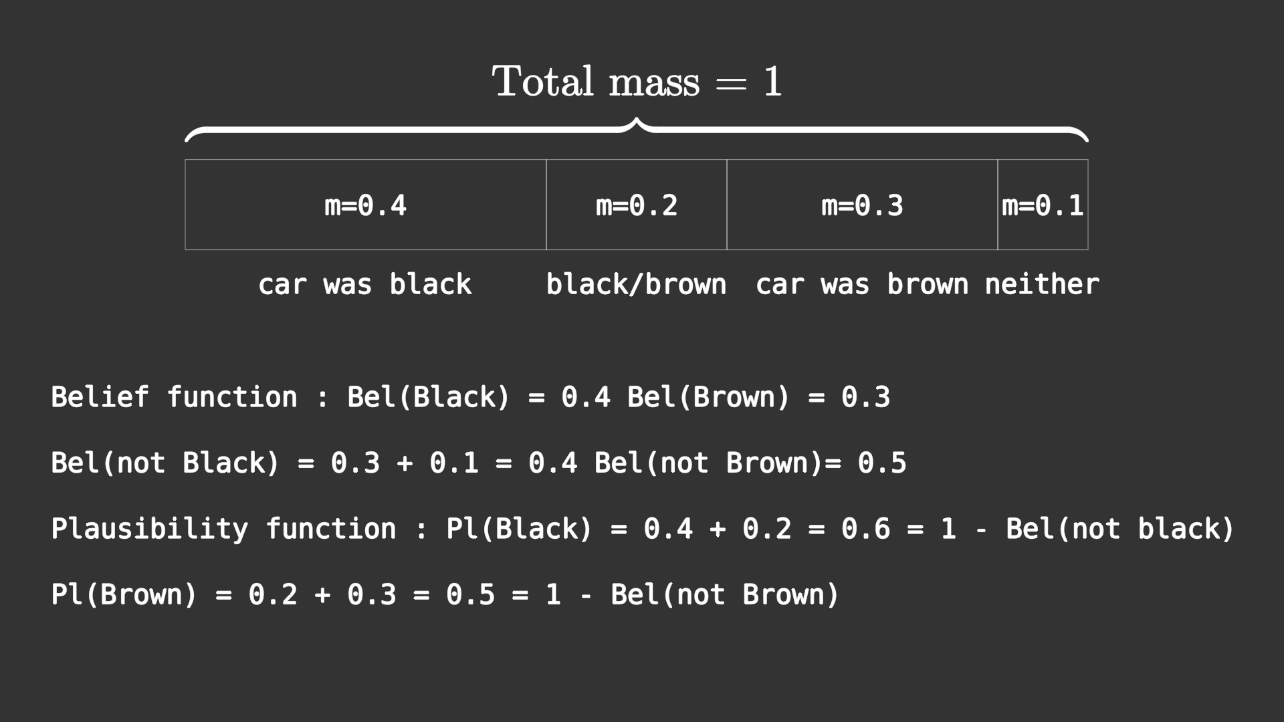


图4‑15信念量计算举例的效果图

2.证据决策树整体算法的思路图（图4‑16）

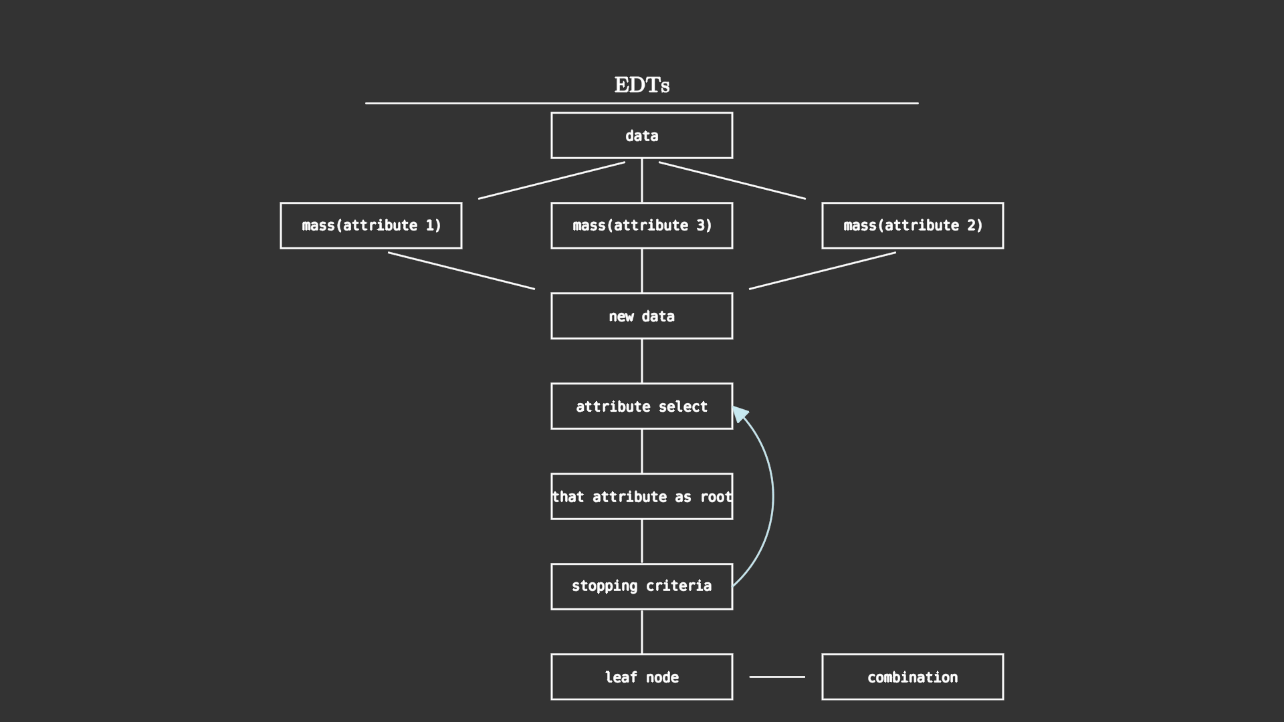
在动画中我们绘制了证据决策树算法的思路逻辑图，从数据导入开始，直到最后的多个叶节点的融合过程。本文的整体实现算法基本按照如图中所示进行。

图4‑16证据决策树整体思路的效果图

3.证据决策树的构造过程（图4‑17）

下图展示了证据决策树第一次叶节点的构造过程。红圈表示训练集实例，方框表示节点，各个箭头表示该节点下的分支情况，即自子节点。箭头末端为当前子节点的名称，名称下方为该分支标记。从图中可见当前分支过程使用的是abiltiy属性，产生了8个子节点，各个训练集实例分别按照前文所规定的进入各自节点。图中较暗的分支表明该分支为空，将会被修剪删除。

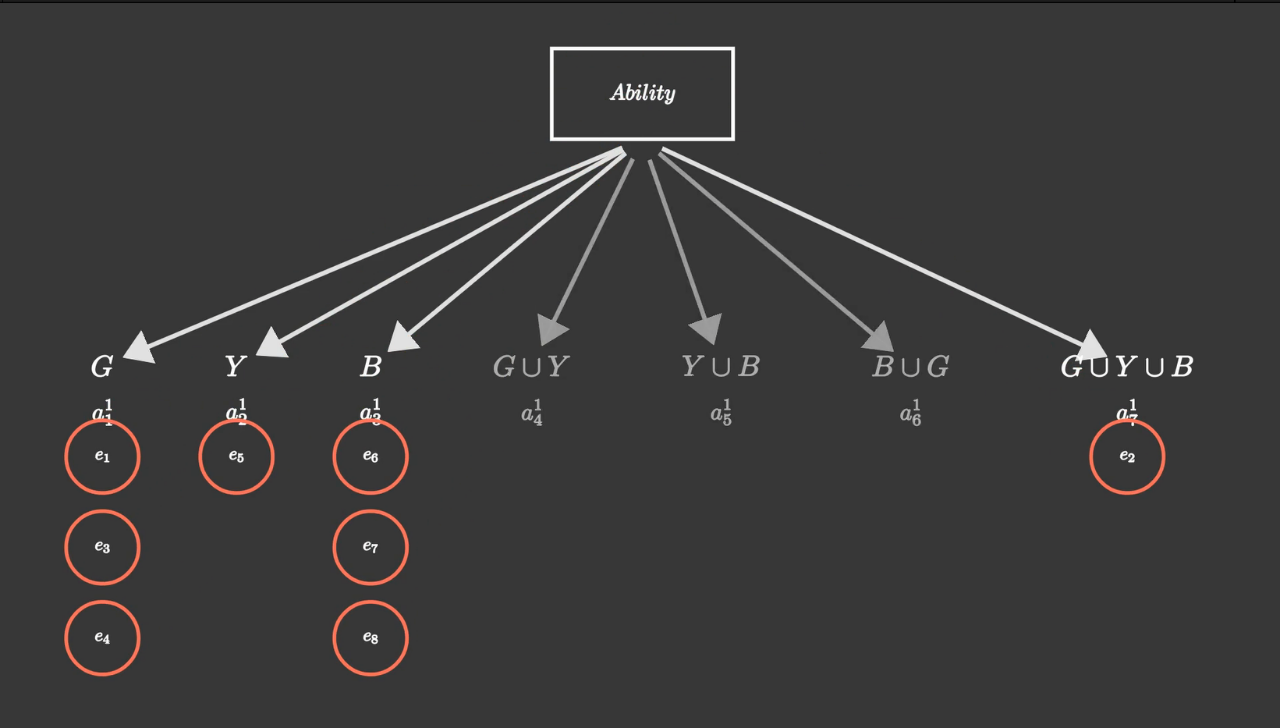


图4‑17证据决策树的构造过程的效果图

4.证据决策树对新数据实例的预测（图4‑18）

下图为在整棵证据决策树构造完成后，我们对新实例（）进行预测的过程。左上角为新实例的各个属性的质量函数表达。经过分类运算之后，可以看到代表新实例的红圈到达了两个叶节点，这是区别于传统机器学习中的决策树的。随后我们的动画将展示对个两个节点的证据进行融合的过程。

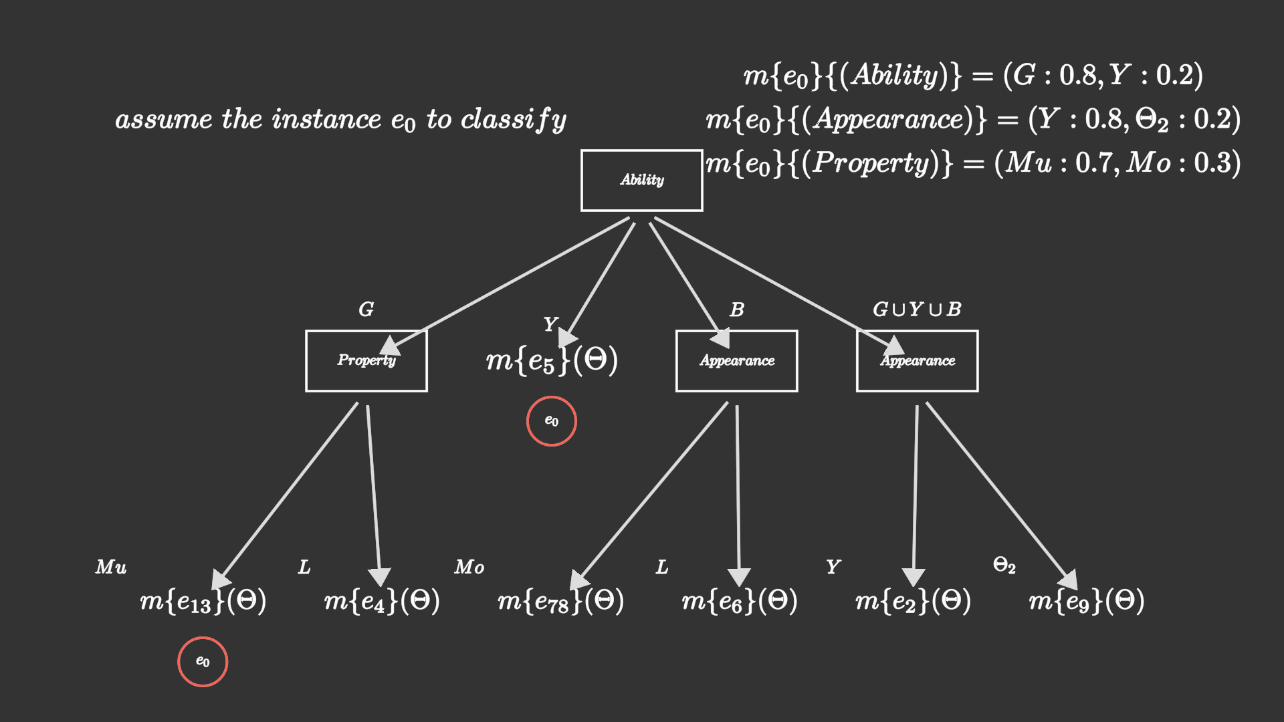


图4‑18证据决策树的预测过程的效果图

### 提供的 API 函数接口

为了便于后续继续的开发利用，我们提供了两个可以生产动画的API函数接口。其一是基于C4.5决策树算法的，只需传入模型，即可产生一段树结构生成的动画效果。生成的树的形状由决策树算法和传入的数据集决定，与算法的模型训练过程完全一致。其将首先展示所生成树的整体构造，随后可以放大查看各个节点内部的分类标准，以及叶节点所代表的的类标签。

下面是传入鹫尾花数据集后生成的树的动画的部分截图（图4‑19、图4‑20）：

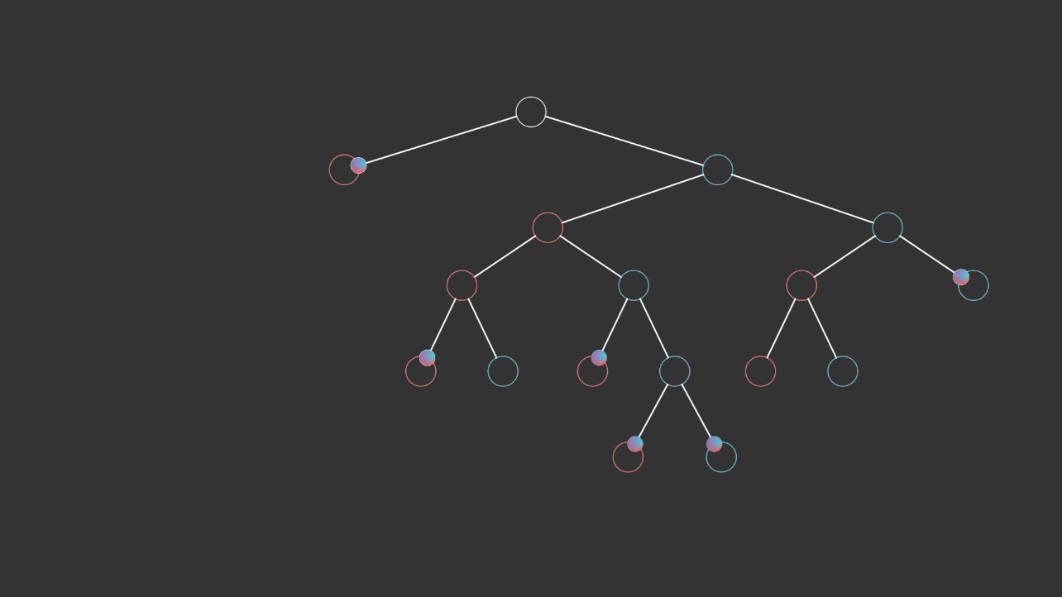


图4‑19决策树的效果图

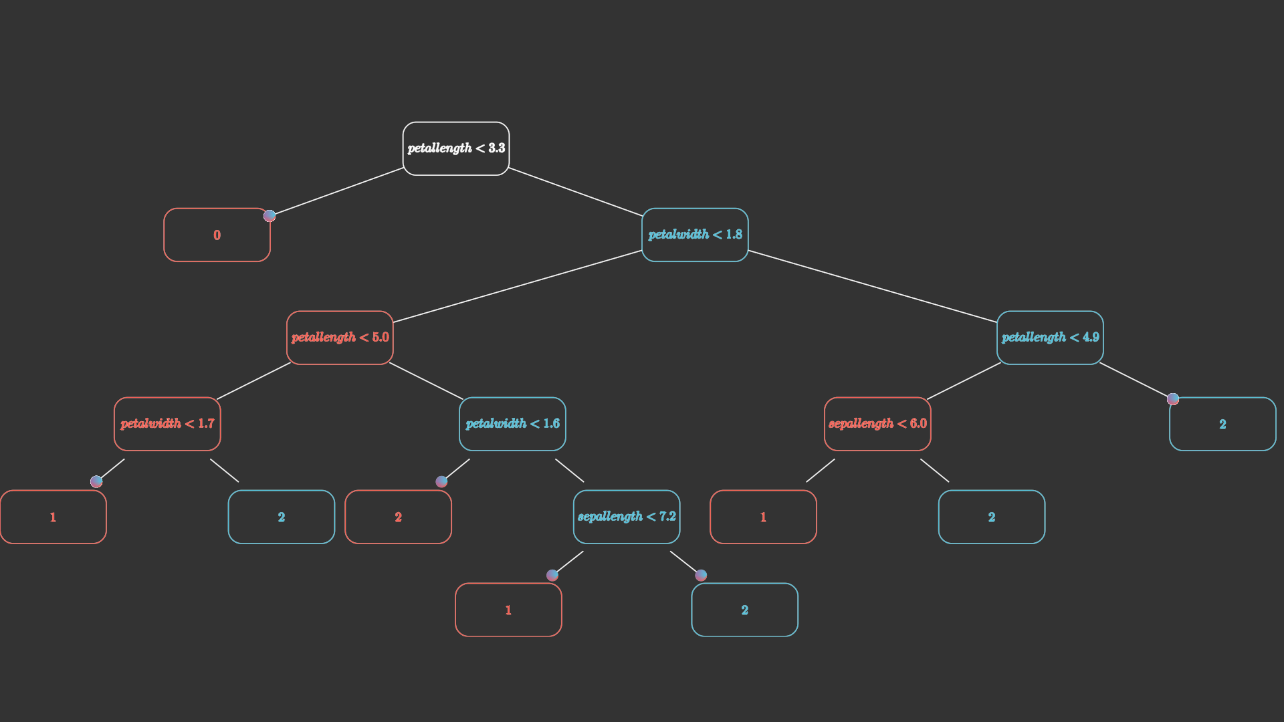
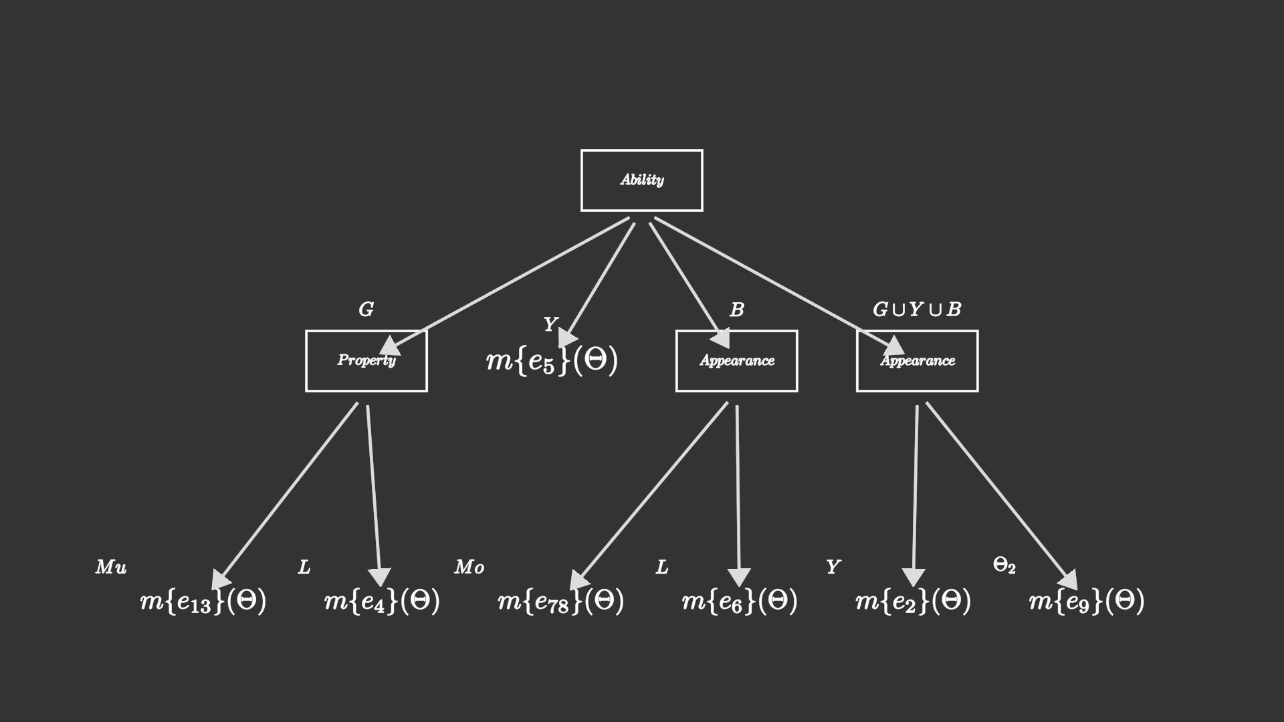


图4‑20完整显示的决策树效果图

其二是基于第三章中所介绍的证据决策树算法的，同样只需传入模型。下面以本文第三章中数据集生成的树的动画的截图（图表 4‑21）：



图表 4‑21证据决策树的效果图

## 本章小结

本章介绍了使用Manim制作基于证据学习的决策树的动画视频的过程，阐述了部分概念的动画设计思路以及整体视频的设计逻辑。接着解释了动画的开发渲染过程。最后展示了部分程序代码执行后的成品效果图。

# 总结与展望

本章总结了全文，陈述了本文的主要工作，归纳了创新点。并且在本文的基础上展望了之后的可能方向。

## 本文总结

本文针对目前没有证据决策树可视化的工具的现状，设计开发了一个基于Manim数学动画引擎的证据决策树的可视化框架，兼顾介绍证据理论的一些基础概念。为初学者理解证据学习领域提供一个直观感受，建立对证据理论的大致知识体系。

### 本文的主要工作

本文主要研究了如何将证据决策树运行逻辑进行可视化操作，主要工作内容有以下：

1. 完成了证据决策树的算法的实现，能够进行简单模型训练和预测功能，能够完整记录训练和预测过程中的数学变化；
2. 完成了质量函数的数据结构，能完成对数据集的读取存储，能进行基本证据操作；
3. 设计了基于Manim数学动画引擎的证据决策树的运行逻辑的视频动画，探索了可视化证据学习的方向；
4. 编写了两个提供决策树和证据决策树的动画生成的API接口，方便以后再次利用开发；
5. 为初学者提供了视频方式了解证据学习领域的途径。

### 本文的主要创新点

本文所设计的动画效果的创新点有以下几点：

1. 使用开源动画引擎Manim，便于开发扩展，且基于Python库，易于与机器学习代码结合；
2. 同时兼顾训练预测和可视化的过程，在训练的同时以动画展示；
3. 部分动画打包成接口，为后续二次开发提供便利；
4. 提供了证据学习领域可视化的可能方向；
5. 目前机器学习领域的可视化多为静态图，本次可视化使用动图的形式，更能体现算法的连续性，更具有解释性。

## 展望

本段从证据决策树可视化的意义和可能应用方向入手，简单叙述了其未来潜力。随后，将讨论其存在的不足，并针对其劣势，提出了可能的解决方向。

### 本文的应用场景

本文的可视化的意义及主要应用场景如下：

1. 探索了证据学习领域的可视化的途径，能为初学者入门该领域提供简单帮助，或者吸引他人对该领域的兴趣；
2. 可在未来发展成为以一个针对证据学习领域算法可视化的Python库；
3. 可以应用于教学场景，为教师提供 一个教学的手段，便于阐释理论的内涵。

### 本文的不足及改进措施

本文只是简单的尝试了证据决策树的可视化工作，其仍然存在许多不足之处：

1. 本文仅选取了一种证据决策树的算法作为底层构造算法，并没有兼顾其它如信念证据决策树、模糊证据决策树等。因此，有待后来者进行补充拓展。此外，本文仅实现了必要的证据理论所需的函数，并没有包含所有，具有一定的局限性。
2. 出于作者水平有限，对本文概念的可视化并非完全严谨，可能存在理解和阐述上的偏差。
3. 受限于Manim的渲染机制，无法实现一些特殊的图形的绘制，以及一些多进程的动画效果制作。
4. 基于上述情况，在日后的研究中应当更加加强对该动画引擎的源码研究和证据理论学习，配合FFmpeg实现更完整更全面的证据决策树的可视化框架。

# 致谢

旧章之末，新篇伊始。时间总是无情地推动着我们向前，在匆忙中我的大学生活也迎来了谢幕。我要感谢上海大学，给予了我学习与成长的舞台。感谢中欧学院，以丰富多样的课程开拓了我的眼界。其中的机器学习与数据工程两门课对我影响巨大，各位老师以自己丰富的阅历带我进入了大数据与人工智能的领域。

感谢我的导师陈伦德老师和杜水淼老师。两位老师对我的毕设提供了巨大的支持与帮助，同时在多门课程中，两位老师不厌其烦的传授了我们许多专业知识。

当然我还要感谢我的家人，感谢他们始终对我无论对错的任何选择的支持。

人生如两枚落叶，相逢在匆匆瞬间。这里我要感谢518寝室三位大佬陈斌、闫闵睿和徐海翔对本毕设的犀利建议。感谢工训中心韦、李和杨三位老师对我工程和项目能力的培养。

当身边的一道道风景，变成了回忆，才忽然发现，风景依然在，而人已非少年。

感谢。

# 参考文献

1. Dempster, Arthur P. "Upper and lower probabilities induced by a multivalued mapping." *Classic works of the Dempster-Shafer theory of belief functions*. Springer, Berlin, Heidelberg, 2008. 57-72.
2. Shafer, Glenn. "A mathematical theory of evidence." *A mathematical theory of evidence*. Princeton university press, 1976.
3. 马荔瑶. 信任函数建模的认知不确定性数据分析与学习[D]. 中国科学技术大学, 2016.
4. Delmotte, Francois, and Philippe Smets. "Target identification based on the transferable belief model interpretation of Dempster-Shafer model." *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 34.4 (2004): 457-471.
5. Guo, Huawei, Wenkang Shi, and Feng Du. "EDTs: Evidential decision trees." *International Conference on Fuzzy Systems and Knowledge Discovery*. Springer, Berlin, Heidelberg, 2005.
6. Jousselme, Anne-Laure, Dominic Grenier, and Éloi Bossé. "A new distance between two bodies of evidence." *Information fusion* 2.2 (2001): 91-101.
7. Elouedi, Zied, Khaled Mellouli, and Philippe Smets. "Belief decision trees: theoretical foundations." *International Journal of Approximate Reasoning* 28.2-3 (2001): 91-124.
8. Polpitiya, Lalintha G. *A Framework for Efficient Implementation and Effective Visualization of Dempster-Shafer Belief Theoretic Computations for Reasoning Under Uncertainty*. Diss. University of Miami, 2019.
9. Sammon, John W. "A nonlinear mapping for data structure analysis." *IEEE Transactions on computers* 100.5 (1969): 401-409.
10. Fenoeux, T., and M. Skarstein Bjanger. "Induction of decision trees from partially classified data using belief functions." *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics.'cybernetics evolving to systems, humans, organizations, and their complex interactions'(cat. no. 0*. Vol. 4. IEEE, 2000.
11. Sutton-Charani, Nicolas, Sébastien Destercke, and Thierry Denœux. "Classification trees based on belief functions." *Belief Functions: Theory and Applications*. Springer, Berlin, Heidelberg, 2012. 77-84.
12. Elouedi, Zied, Khaled Mellouli, and Philippe Smets. "Decision trees using the belief function theory." *Proceedings of the international conference on Information Processing and Management of Uncertainty IPMU*. Vol. 1. 2000.
13. Elouedi, Zied, Khaled Mellouli, and Philippe Smets. "Belief decision trees: theoretical foundations." *International Journal of Approximate Reasoning* 28.2-3 (2001): 91-124.
14. Denœux, Thierry, and M-H. Masson. "EVCLUS: evidential clustering of proximity data." *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 34.1 (2004): 95-109.
15. Polpitiya, Lalintha G. *A Framework for Efficient Implementation and Effective Visualization of Dempster-Shafer Belief Theoretic Computations for Reasoning Under Uncertainty*. Diss. University of Miami, 2019.
16. Denoeux, Thierry. "Decision-making with belief functions: a review." *International Journal of Approximate Reasoning* 109 (2019): 87-110.
17. Tong, Zheng, Philippe Xu, and Thierry Denoeux. "An evidential classifier based on Dempster-Shafer theory and deep learning." *Neurocomputing* 450 (2021): 275-293.
18. Reineking, Thomas. *Belief functions: theory and algorithms*. Diss. Universität Bremen, 2014.
19. Kotsiantis, Sotiris B. "Decision trees: a recent overview." *Artificial Intelligence Review* 39.4 (2013): 261-283.
20. Breiman, L. , et al. "Classification and Regression Trees (CART)." Biometrics 40.3(1984):358.
21. Trabelsi, Asma, Zied Elouedi, and Eric Lefevre. "Decision tree classifiers for evidential attribute values and class labels." *Fuzzy Sets and Systems* 366 (2019): 46-62.
22. Chen, Qi, et al. "Data classification using the Dempster–Shafer method." *Journal of Experimental & Theoretical Artificial Intelligence* 26.4 (2014): 493-517.