第一章 绪论

Lianghai Xiao

https://github.com/styluck/mlb

作业邮箱：alswfx@126.com

什么是案例实物课?

- 其实就是统计分析+实际案例
- **收集、处理、分析和解释数据的方法**
- **数据收集：**计实验或调查，获得数据
- **数据处理：**

  清洗：识别并纠正（或删除）数据中的错误、重复或不完整的记录。

  转换：将数据转换成适合分析的格式，可能包括标准化数据、编码分类变量或将数据转换为不同的数据类型。

  整合：合并来自不同来源的数据集，以创建一个统一的数据集，这可能涉及解决数据不一致性。

  降维：减少数据集中变量数量的技术，目的是简化模型，提高计算效率，同时尽量保留原始数据的重要信息。

什么是案例实物课?

- **数据分析和解释：**
- 数据的探索、处理、分析和得出结论。这个过程可以帮助我们理解数据背后的含义，发现模式、趋势和关联，以及做出基于数据的决策。常用的手段包括：
- **数据可视化：**使用图表和图形来展示数据分析的结果。
- **相关性分析**：评估变量之间的线性关系，如皮尔逊相关系数。
- **回归分析**：用于预测和评估变量之间的关系，包括线性回归、逻辑回归、时间序列分析等。
- **分类和聚类**：将数据分为不同的类别或组，如决策树、K-means聚类等。
- **机器学习**：应用各种算法来发现数据中的模式和关系，如支持向量机、神经网络等。

课程安排

- **总体目标：**
  - 掌握运用python和统计方法设计处理实际应用问题的软件工具的能力，具备应用软件工具开发处理实际问题的能力。
- **课程要求：**
  - 上课、试验作业
- 成绩计算方法：
  - 平时成绩30%：上课和试验出勤10%，作业和实验20%
  - 期末成绩70%：期末论文

课程安排

- 参考教材（非必要不购买）：
  - 王小川、史峰、郁磊、李阳，MATLAB神经网络43个案例分析，北京航空航天大学出版社，2013年。
  - 谢中华，MATLAB统计分析与应用：40个案例分析（第二版），北京航空航天大学出版社，2015年。
  - 李华、袁先智、赵建彬，金融科技大数据风控方法介绍，科学出版社，2023年。
  - 邱锡鹏，神经网络与深度学习，机械工业出版社，2022年。
- 参考资料：
  - 李东风，多元统计分析讲义：https://www.math.pku.edu.cn/teachers/lidf/course/mvr/mvrnotes/html/_mvrnotes/index.html
  - 何志坚，数理统计讲义：https://bookdown.org/hezhijian/book/

课程安排

- 课件资料分享网站:
- https://github.com/styluck/mlb

- 作业邮箱:
- alswfx@126.com

统计法基础知识

- 教材及参考资料
  - 中华人民共和国统计法，中国法制出版社
  - https://www.gov.cn/zhengce/content/2017-06/19/content_5203711.htm
  - 中华人民共和国预算法、中华人民共和国统计法（中英对照），法律出版社
  - 安建主编，中华人民共和国统计法释义，中国法制出版社

统计法基础知识

- 总体目标:
  - 掌握统计法条目内容
- 成绩计算方法:
  - 期末论文100%

分析的工具

- **MATLAB：**
- 优势：Matlab是为数值计算和矩阵操作设计的，因此在这些领域具有很高的效率。拥有大量的内置数学函数和工具箱，可以快速实现复杂的数学运算。
- 劣势：Matlab是商业软件，需要购买许可证，成本较高。代码通常依赖于Matlab环境，跨平台运行可能需要额外的工作。Matlab不是开源的，限制了代码的共享和定制化。

分析的工具

- **Python：**
- 优势：Python是开源的，可以免费使用。Python是一种通用编程语言，适用于多种编程任务，包括Web开发、自动化脚本等。拥有大量的第三方库，如NumPy、SciPy、Pandas、Matplotlib、Scikit-learn等。
- 劣势：Python的执行速度通常不如Matlab，特别是在数值计算密集型任务上。Python通常需要多个库和工具的组合来实现Matlab中的某些功能。

Python编程简介

- 先安装Anaconda
- 下载地址
  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/Anaconda3-2020.11-Windows-x86_64.exe
  - 安装的时候需要打勾的地方就打勾

- python版本：3.8.5
- pandas版本：1.1.3　　pip install pandas==1.1.3
- numpy版本：1.22.0　　pip install numpy==1.22.0

本课程中三个最常用的包

- import numpy as np
- import pandas as pd
- import matplotlib.pyplot as plt

基本数据结构

- **列表（List）**：有序的可变集合，元素可以是不同类型。
- lst = [1, 2, 3, "a", "b"]
- **元组（Tuple）**：有序的不可变集合，元素可以是不同类型。
- tpl = (1, 2, 3, "a", "b")
- **集合（Set）**：无序的唯一元素集合，用于去重和集合运算。
- st = {1, 2, 3, 4}
- **字典（Dictionary）**：键值对的集合，键是唯一的，值可以是任何类型。
- dct = {"a": 1, "b": 2, "c": 3}

控制结构

- **条件语句：**if、elif、else 用于控制程序执行的不同路径。
- if x > 0:
- print("Positive")
- elif x == 0:
- print("Zero")
- else:
- print("Negative")

控制结构

- **for 循环**：用于迭代序列（如列表、元组、字符串等）。
- for i in range(5):
- print(i)
- **while 循环**：根据条件反复执行代码块。
- while condition:
- print("Looping")

列推导式

- **列推导式**：简洁地生成列。
- squares = [x**2 for x in range(5)].

- **生成器表达式**：生成器是一种惰性迭代器，不会一次性生成所有值，节省内存。
- gen = (x**2 for x in range(5))

函数与类

- **函数（Function）**：通过 def 关键字定义，封装可复用的代码。
- def add(a, b):
- 	return a + b
- **类（Class）**：面向对象编程中的基本结构，通过定义类来组织数据和行为。
- class Dog:
- 	def __init__(self, name):
- 		self.name = name
- 	def bark(self):
- 		print(f“How are you doing {self.name} ?")

异常处理结构

- **try / except**：用于处理异常和错误，保证程序不崩溃。
- **finally**：无论是否发生异常，finally 中的代码总是会执行。
- try:
- x = 1 / 0
- except :
- print("Error")
- finally:
- print("This will always run")

上下文管理器

- 上下文管理器（Context Manager）：通过 with 关键字管理资源的获取和释放。
- with open("file.txt", "r") as file:
- 	content = file.read()

- with open("example.txt", "w") as file:
- 	file.write("Hello, World!")

装饰器

- 装饰器（Decorator）：修改或扩展函数行为的高阶函数。
- def my_decorator(func):
- 	def wrapper(*args, **kwargs):
- 		print(f"Function '{func.__name__}' was called with arguments: {args} {kwargs}")
- 		result = func(*args, **kwargs)
- 		print("After function")
- 		return result
- 	return wrapper

numpy的广播机制（Broadcasting）

- 允许 numpy 在算术操作中自动扩展较小的数组，使其与较大的数组具有相同的形状，以便它们可以一起进行元素级操作，而无需显式地复制数据。
- **广播机制的基本规则**
  - 如果两个数组的维度数不相同，那么在较小的数组形状前面补充维度1，直到两个数组的维度数相同。
  - 如果两个数组在某个维度上的大小不同，但其中一个数组在该维度上的大小为1，那么该数组在这个维度上被扩展为与另一个数组相同的大小。
  - 如果在任何维度上两个数组的大小都不同并且都不为1，那么这两个数组就不兼容，不能进行广播操作。

示例 1：标量与数组的运算

- import numpy as np

- array = np.array([1, 2, 3])
- result = array + 10
- print(result)

- 输出
- [11 12 13]

示例 2：二维数组与一维数组的运算

- matrix = np.array([[1, 2, 3], [4, 5, 6]])
- vector = np.array([1, 2, 3])

- result = matrix + vector
- print(result)

- 输出：
- [[2 4 6]
- [5 7 9]]

示例 3：广播失败的情况

- matrix = np.array([[1, 2, 3], [4, 5, 6]])
- vector = np.array([1, 2])

- # 试图进行广播运算
- result = matrix + vector

示例 3：广播失败的情况

- matrix = np.array([[1, 2, 3], [4, 5, 6]])
- vector = np.array([1, 2])

- vector_reshaped = vector[:, np.newaxis]

- 输出：
- [[2 3 4]
  [6 7 8]]

pandas练习数据下载

- https://jnueducn-my.sharepoint.com/:f:/g/personal/xiaolh_jnu_edu_cn/EgcxNETYfN1BgeptTyhO1IoBCAHLy0AMMt7zuiREDAL-uQ?e=g4bstp

Pandas包

- Pandas 是 Python 数据分析中非常流行的库，它提供了强大、灵活且易于使用的数据处理和分析工具，用于高效地操作表格型和时间序列数据。
- pandas 的主要**数据结构**:
- **Series**：一维的数据结构，类似于带标签index的数组或列表。
- **DataFrame**：二维的表格型数据结构，类似于电子表格或 SQL 表。
- index column

Pandas包

- **数据加载与存储:**
- 自带函数，方便读取和写入数据，例如 CSV、Excel、SQL 数据库、JSON 等。
- **数据清洗:**
- 自带函数，方便处理缺失值（填充、删除等）。数据过滤、排序和重塑。重命名列、索引操作等。

Pandas包

- 查看 DataFrame 的前几行
- print(df.head())
- 查看 DataFrame 的基本信息（列名、数据类型、内存使用等）
- print(df.info())
- 查看统计汇总信息
- print(df.describe())

选择与筛选数据

- 选择某列
- print(df['time'])
- 选择多列
- print(df[['time', 'close']])
- 按条件筛选
- df_filtered = df[df['Age'] > 30]
- 按索引行选择
- print(df.iloc[0]) # 第一行
- print(df.loc[0]) # 使用标签选择（如果存在标签）

处理缺失值

- 检查缺失值
- print(df.isna().sum())

- 填充缺失值
- df.fillna(0, inplace=True)

- 删除含有缺失值的行
- df.dropna(inplace=True)

数据类型转换

- 转换某列的数据类型
- df['close'] = df['close'].astype(float)

- 将 DataFrame 转换为 NumPy 数组
- data_array = df.values