复习题

1、PageRank的设计思想是什么?

PageRank的设计思想是基于网页之间的链接关系来衡量网页的重要性。它认为，如果一个网页有很多其他网页链接到它，那这个网页可能更重要。而且，来自重要网页的链接会比来自普通网页的链接更有“权重”。

这个算法的目的是通过计算每个网页的链接情况，来判断哪些网页在整个网络中更有影响力，进而在搜索结果中排得更靠前。

2、贝叶斯定理的内容是什么?它又有哪些重要应用?

贝叶斯定理是用来计算在已知条件下某个事件发生的概率，核心思想是通过已有的信息更新对事件发生概率的估计。根据查阅的资料，贝叶斯定理的公式如下：

P(A∣B)=P(B∣A)⋅P(A)P(B)P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}P(A∣B)=P(B)P(B∣A)⋅P(A)

在这个公式中：

- P(A∣B)P(A|B)P(A∣B) 是在事件 B 发生时，事件 A 发生的概率。
- P(B∣A)P(B|A)P(B∣A) 是在事件 A 发生时，事件 B 发生的概率。
- P(A)P(A)P(A) 是事件 A 发生的初始概率。
- P(B)P(B)P(B) 是事件 B 发生的概率。

贝叶斯定理的作用在于，它允许我们根据新获取的信息更新对事件的看法。

贝叶斯定理有许多重要应用。比如，在**医疗诊断**中，可以计算检测结果呈阳性时，患者实际患病的概率，这对医生的判断非常有帮助。另一个例子是**机器学习**，尤其是在朴素贝叶斯分类器中，它被广泛应用于分类任务，比如垃圾邮件识别。

在**金融和经济学**中，贝叶斯定理常用于处理不确定性，帮助评估市场走势和投资风险。此外，在**人工智能**领域，贝叶斯网络等工具利用这个定理来处理复杂的概率推断问题。

3、试阐述蒙特卡罗方法的基本原理。

蒙特卡罗方法是一种基于随机抽样的数值计算技术，广泛应用于解决复杂的数学问题和进行概率模拟。这种方法的核心在于使用随机数生成器，随机生成大量数据点以代表整个样本空间。这种方式的优势在于能够在无法精确计算的情况下，利用“通过随机生成的数据，近似估计所需的量”。

在具体应用中，蒙特卡罗方法常用于积分、优化和统计推断等问题。例如，在求解某个函数的定积分时，生成大量随机点以落在该函数的定义域内，并计算这些点落在函数下方的比例，以此来近似估计积分的值。

根据一些文献，蒙特卡罗方法在不确定性分析和风险评估中也被广泛采用。在金融领域，“该方法可以模拟资产价格的变化，评估投资组合的风险和收益”，提供了一种有效的风险管理工具。

此外，蒙特卡罗方法在物理学、工程学以及计算机科学等多个领域中也有应用，尤其是在粒子物理和流体动力学的模拟研究中，证明了其强大的适应性和实用性。

4、梯度下降法的主要思想是什么?你能用通俗的语言解释出来吗?

梯度下降法是一种优化算法，常用于机器学习和深度学习中，主要目标是最小化损失函数。它的基本思想可以用通俗的语言来描述。我们可以想象自己站在一个山谷的边缘，周围的地形崎岖不平。我们希望找到山谷的最低点，也就是最优解。首先，从一个随机选择的起始点开始，然后计算这个点的梯度。梯度指向了函数上升最快的方向，就像站在山顶时，指向最高处的箭头。接下来，我们需要沿着梯度的相反方向移动，也就是往下坡的方向走。这一步的大小由一个叫学习率的参数决定，学习率就像是你每次下坡的步伐大小。步伐太大可能会让你跌倒，步伐太小则会让你走得很慢。通过不断重复这个过程，计算新的点的梯度并调整位置，我们会逐渐接近山谷的底部，找到函数的最低点，也就是损失函数的最小值。最终，我们的目标是找到最佳参数，使得模型的预测更加准确。虽然梯度下降法简单易用，但在实际应用中需要注意选择合适的学习率，以避免陷入局部最优解。

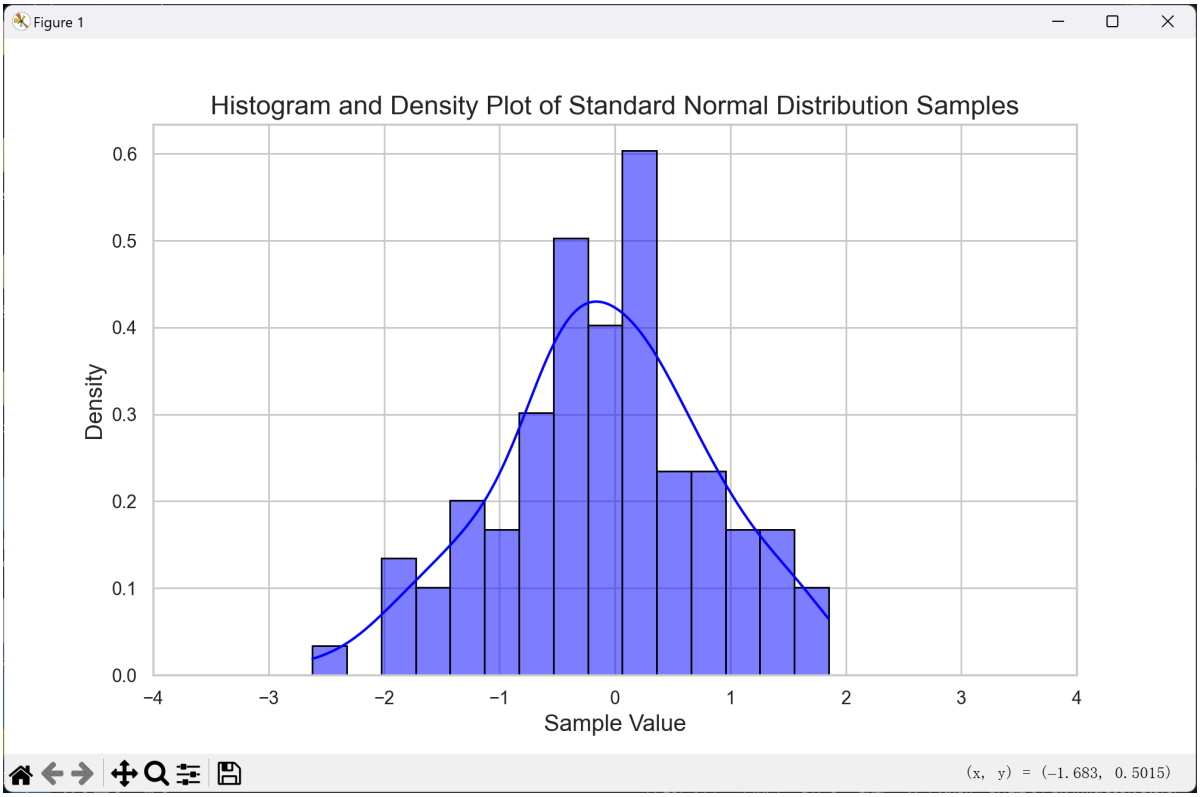
实践题

1、使用 numpy 生成服从标准正态分布的 100个样本。

```python
import numpy as np

# 设置随机种子以便复现结果
np.random.seed(42)

# 生成100个服从标准正态分布的样本
samples = np.random.normal(loc=0, scale=1, size=100)

# 打印生成的样本
print(samples)
```

```
[ 0.49671415, -0.1382643 ,  0.64768854,  1.52302986, -0.23415337, -0.23413696,
1.57921282,  0.76743473, -0.46947439,  0.54256004, -0.46341769, -0.46572975,
0.24196227, -1.91328024, -1.72491783, -0.56228753, -1.01283112,  0.31424733,
-0.90802408, -1.4123037 ,  1.46564877, -0.2257763 ,  0.0675282 , -1.42474819,
-0.54438272,  0.11092259, -1.15099358,  0.37569802, -0.60063869, -0.29169375,
-0.60170661,  1.85227818, -0.01349722, -1.05771093,  0.82254491, -1.22084365,
0.2088636 , -1.95967012, -1.32818605,  0.19686124,  0.73846658,  0.17136828,
-0.11564828, -0.3011037 , -1.47852199, -0.71984421, -0.46063877,  1.05712223,
0.34361829, -1.76304016,  0.32408397, -0.38508228, -0.676922  ,  0.61167629,
1.03099952,  0.93128012, -0.83921752, -0.30921238,  0.33126343,  0.97554513,
-0.47917424, -0.18565898, -1.10633497, -1.19620662,  0.81252582,  1.35624003,
-0.07201012,  1.0035329 ,  0.36163603, -0.64511975,  0.36139561,  1.53803657,
-0.03582604,  1.56464366, -2.6197451 ,  0.8219025 ,  0.08704707, -0.29900735,
0.09176078, -1.98756891, -0.21967189,  0.35711257,  1.47789404, -0.51827022,
-0.8084936 , -0.50175704,  0.91540212,  0.32875111, -0.5297602 ,  0.51326743,
0.09707755,  0.96864499, -0.70205309, -0.32766215, -0.39210815, -1.46351495,
0.29612028,  0.26105527,  0.00511346, -0.23458713]
```

2、通过 Python 程序为抽样出的样本绘图展示




```python
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# 设置随机种子以便复现结果
np.random.seed(42)

# 生成100个服从标准正态分布的样本
samples = np.random.normal(loc=0, scale=1, size=100)

# 设置绘图风格
sns.set(style="whitegrid")

# 绘制直方图和密度图
plt.figure(figsize=(10, 6))
sns.histplot(samples, bins=15, kde=True, stat='density', color='blue',
edgecolor='black')

# 添加标题和标签
plt.title('Histogram and Density Plot of Standard Normal Distribution Samples',
fontsize=16)
plt.xlabel('Sample Value', fontsize=14)
plt.ylabel('Density', fontsize=14)
plt.xlim(-4, 4)  # 设置x轴范围

# 显示图形
plt.show()
```

3、通过 Python 程序计算矩阵 (2145)(2415) 的特征值和特征向量。

```python
import numpy as np

# 定义矩阵
matrix = np.array([[2, 1],
                   [4, 5]])

# 计算特征值和特征向量
eigenvalues, eigenvectors = np.linalg.eig(matrix)

print("特征值:", eigenvalues)
print("特征向量:\n", eigenvectors)
```

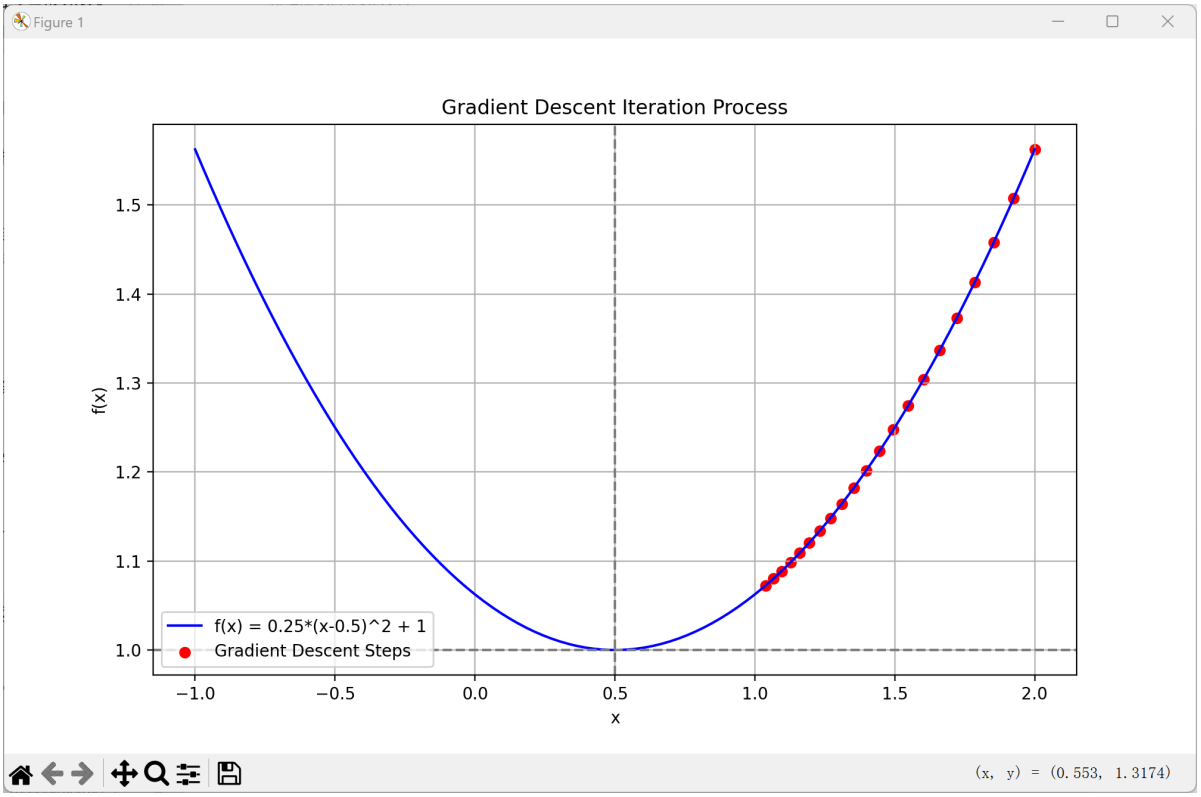
```
PS C:\Users\13803\Desktop\大二作业\数据科学\代码> & C:/Users/13803/Ap
cal/Microsoft/WindowsApps/python3.12.exe c:/Users/13803/Desktop/大二作
科学/代码/6.py
特征值: [1. 6.]
特征向量:
 [[-0.70710678 -0.24253563]
 [ 0.70710678 -0.9701425 ]]
```

5、

```python
import numpy as np

# 定义数据矩阵
data_matrix = np.array([[1, 2, 3],   # 第一行：data
                        [1, -1, 4],  # 第二行：X
                        [2, 1, 3],   # 第三行：Y
                        [1, 3, -1]])  # 第四行：Z

# 计算协方差矩阵
cov_matrix = np.cov(data_matrix, rowvar=False)  # rowvar=False 表示每列代表一个变量

# 打印协方差矩阵
print("协方差矩阵:\n", cov_matrix)
```

```
协方差矩阵:
 [[ 0.25        -0.08333333  0.25      ]
 [-0.08333333  2.91666667 -3.08333333]
 [ 0.25       -3.08333333  4.91666667]]
```

附加题




```
cal/Microsoft/WindowsApps/python3.12.exe c:/Users/13803/Desktop/大二作
科学/代码/8.py
局部极小值的位置: x = 1.0377288836128136, f(x) = 1.0722880880678707
```