图标

描述已自动生成

**Python大数据与人工智能试卷实验报告**

|  |  |  |  |
| --- | --- | --- | --- |
| **专业** | **：** | **计算机电脑** |  |
| **班级** | **：** | **CS2002** |  |
| **学号** | **：** | **I201920029-I201920024** |  |
| **姓名** | **：** | **冯就康-木林** |  |
| **电话** | **：** | **15623031879-15623029026** |  |
| **邮箱** | **：** | **sokhorng526@gmail.com-juuzousuzuya077@gmail.com** |  |

目录

[I. 简介 3](#_Toc153803482)

[1. 任务的简介 3](#_Toc153803483)

[2. 电影评分预测问题的简介 3](#_Toc153803484)

[II. 模型的概述 4](#_Toc153803485)

[III. 模型的详细设计 5](#_Toc153803486)

[1. 数据预处理 5](#_Toc153803487)

[1.1 数据分析与预处理 5](#_Toc153803488)

[1.2 获取电影属性的额外信息 10](#_Toc153803489)

[2. 神经网路模型的设计 12](#_Toc153803490)

[2.1 数据预处理: 12](#_Toc153803491)

[2.2 模型的设计: 12](#_Toc153803492)

[2.3 模型的编译 13](#_Toc153803493)

[2.4 训练循环 14](#_Toc153803494)

[2.5 模型评估 14](#_Toc153803495)

[2.6 调优模型参数: 14](#_Toc153803496)

[IV. 模型的评估 16](#_Toc153803497)

[V. 实验总结与思想 17](#_Toc153803498)

[1. 实验总结 17](#_Toc153803499)

[2. 实验思想 17](#_Toc153803500)

# 简介

## 任务的简介

**任务**：根据电影属性，预测电影星级评分

**基本内容**：

* 将9742部电影随机划分训练集(data\_train)与验证集（data\_val）
* 使用训练集训练模型，预测验证集电影的标签
* 重复三次“训练集/验证集划分、训练、测试”，汇报三次的平均准确率
* 数据集划分：

from sklearn.model\_selection import train\_test\_split

data\_train, data\_val = train\_test\_split（data, test\_size=0.1）

## 电影评分预测问题的简介

电影评分预测是推荐系统领域的一个重要问题。它涉及到根据用户过去的行为和偏好，预测用户对他们尚未看过的电影会给出的评分。这个问题的重要性有几个原因：

* **个性化**：通过准确预测电影评分，平台可以提供个性化的推荐，提高了用户的体验。
* **商业价值**：好的推荐能驱动用户的参与度，导致平台使用率和潜在增长平台或公式的收入。
* **资源优化**：它可以帮助用户在大量可用内容中导航，有效地节省他们的时间和精力

# 模型的概述

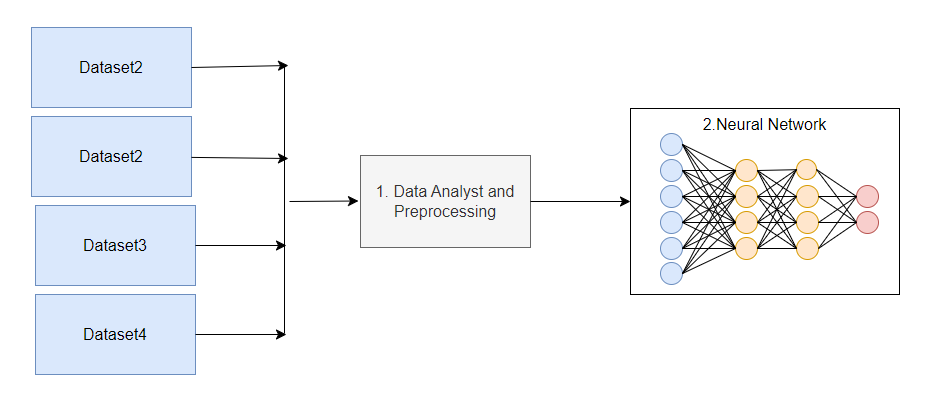
整个模型架构的概述如图1：

图 1 模型架构的概述

模型架构概述的介绍：

1. **数据集**：任何机器学习项目的第一步都是收集和理解数据集。这涉及了解问题域、数据源和数据格式。
2. **数据分析**：收集数据集后，需要对其进行分析以了解其特征。这可能涉及查看值的分布、识别异常值以及理解不同变量之间的关系。
3. **数据分析**：收集数据集后，需要对其进行分析以了解其特征。这可能涉及查看值的分布、识别异常值以及理解不同变量之间的关系。
4. **数据预处理**：原始数据在用于建模之前通常需要进行预处理。这可能涉及清理数据（处理缺失值、删除重复项等）、标准化数据以及将分类变量转换为模型可以使用的格式。
5. **特征工程**：这是创建新特征或修改现有特征以提高模型性能的过程。这可能涉及创建交互项、分箱变量或转换变量。
6. **使用神经网络建模**：数据准备好后，即可用于训练神经网络模型。这涉及定义模型的架构（层数、层类型、激活函数等）、编译模型（定义损失函数、优化器等）以及训练模型。模型训练完成后，可以在验证集上对其进行评估。
7. **模型评估和优化**：使用适当的指标评估模型的性能。如果性能不理想，可以通过调整超参数、更改模型架构或收集更多数据来优化模型。

这是神经网络的高级概述。每个步骤都涉及很多决策，并且可能非常复杂。然而，只要对问题领域和数据有很好的理解，以及在机器学习和神经网络方面的坚实基础，就有可能建立一个在这项任务上表现良好的模型。

# 模型的详细设计

## 数据预处理

本次实验中，我们把数据预处理分成两部分：

* 数据分析与预处理
* 获取电影属性的额外信息

### 数据分析与预处理

数据分析与预处理包含：数据收集、数据集的理解、数据探索性分析、数据质量评估。

* **数据收集**：该过程首先从各种来源（包括数据库和电子表格）收集原始数据。

本次实验我们要使用的数据集是:

Movie Lens Dataset: <https://www.kaggle.com/datasets/aigamer/movie-lens-dataset>

数据集的介绍：

数据集 (ml-latest-small) 描述了电影推荐服务 MovieLens 的 5 星级评级和自由文本标记活动。它包含 9742 部电影的 100836 个评级和 3683 个标签应用程序。这些数据由1996年3月29日至2018年9月24日期间的610名用户创建。该数据集生成于2018年9月26日。

* **数据集的理解**：这涉及理解问题域、数据源和数据格式。确保数据完整、准确且与当前的分析或任务相关的重点。

数据集的主要组成部分包括：

links.csv: 包括movieId、imdbId、tmdbId

movies.csv: 包括movieId、Title、Genres

ratings.csv: 包括userId、movieId、rating、timestamp

tags.csv:包括userId、movieId、tag、timestamp

数据集的概述：

* **userId** --- 为每个用户提供的唯一ID

userId 是随机选择的。他们的 ID 已被匿名化。 ratings.csv 和 tags.csv 之间的用户 ID 是一致的（即，两个文件中相同的 ID 指的是同一用户）。

* **movieId** --- 为每部电影提供的唯一ID

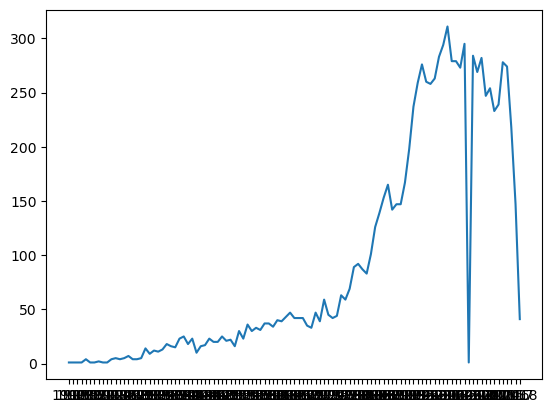
数据集中仅包含至少具有一个评级或标签的电影。这些电影 id 与 MovieLens 网站上使用的一致（例如，id 1 对应于 URL Movie Lens。电影 id 在 ratings.csv、tags.csv、movies.csv 和 links.csv 之间保持一致（即，相同的 id 指的是这四个数据文件中的同一部电影）。

* **rating** --- 评级采用 5 星等级，以半星递增（0.5 星 - 5.0 星）。

所有评级均包含在 ratings.csv 文件中。该文件标题行之后的每一行代表一个用户对一部电影的评分。Mean 显示平均评级为 3.2 星

* **数据探索性分析**：收集数据集后，需要对其进行分析以了解其特征。这可能涉及查看值的分布、识别异常值以及理解不同变量之间的关系。

**数据分析和可视化：**

首先，我们来分析每年年份相应的电影数量：

A screenshot of a computer

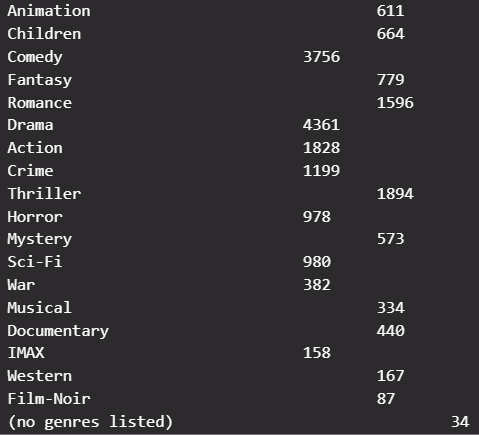
Description automatically generated

这是一个描述2002年电影上映数量的统计摘要。以下是关键信息的解释：

* **最大电影上映数量**：在2002年，最多的电影上映数量为311部。
* **统计数量**（count）：这个数据集包含了107个数据点，也就是说，有107个不同的电影上映的数量被记录下来。
* **平均值**（mean）：平均每年上映约90.93部电影。
* **标准差**（std）：标准差是102.12，这表示电影上映数量的分布在平均值的基础上上下波动约102.12部。
* **最小值**（min）：最少的电影上映数量为1部。
* **25%分位数**（25%）：约25%的年份电影上映数量在16部或小于16部。
* **中位数**（50%）：中位数是39部，意味着一半的年份电影上映数量低于或等于39部，另一半则高于或等于39部。
* **75%分位数**（75%）：约75%的年份电影上映数量在150部或150以下。

其次，我们分析一下哪个电影的类型最受欢迎：

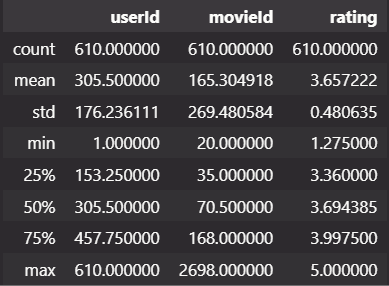
电影类型的数量统计：



这是一个电影类型的分布统计。以下是一些关键的观察结果：

* 喜剧（Comedy）和戏剧（Drama）是数量最多的电影类型，分别有3756部和4361部电影。这可能表明这两种类型的电影最受欢迎，或者制作这两种类型的电影的成本相对较低。
* 动画（Animation）和儿童（Children）\*\*电影的数量相对较少，分别为611部和664部。这可能是因为这些类型的电影需要更多的时间和资源来制作。
* IMAX，西部（Western），黑色电影（Film-Noir）和无类型（no genres listed）的电影数量最少，这可能表明这些类型的电影的需求相对较小，或者制作这些类型的电影的成本相对较高。
* 惊悚（Thriller），动作（Action）和罪案（Crime）电影的数量也相对较多，这可能表明观众喜欢刺激和紧张的情节。

总的来说，这个统计数据提供了关于不同电影类型的数量分布的有用信息，可以帮助我们理解哪些类型的电影更受欢迎，哪些类型的电影可能需要更多的资源来制作等问题。

接下来，我们分析一下电影评分最高的用户以及用户的平均评分：

以下是关键信息的解释：

* **统计数量**（count）：这个数据集包含了610个数据点，也就是说，有610个不同的用户对电影进行了评价。
* **平均值**（mean）：平均的评分是3.66。
* **标准差**（std）：评分的标准差是0.48。这表示评分的分布在平均值的基础上上下波动的程度。
* **最小值**（min）：最低的评分是1.28。
* **25%分位数**（25%）：约25%的评分在3.36或3.36以下。
* **中位数**（50%）：评分的中位数是3.69。这意味着评分低于或等于3.69，另一半则高于或等于这些值。
* **75%分位数**（75%）：约75%的评分在4或4以下。
* **最大值**（max）：最高的评分是5

最后，我们来求一下每部电影的平均评分：

* 计算每部电影的平均评分：首先，按照电影ID(movieId)对评分数据进行分组，然后计算每部电影的平均评分。
* 计算评价每部电影的用户数量：然后，再次按照电影ID对评分数据进行分组，这次计算的是评价每部电影的用户数量。
* 合并电影信息和评分信息：接下来，将电影信息和计算得到的平均评分以及用户数量进行合并，得到一个新的数据框，其中包含每部电影的信息、平均评分和评价用户数量。
* 找出评价电影数量最多的用户和该用户的平均评分：按照用户ID(userId)对评分数据进行分组，计算每个用户评价的电影数量和平均评分，然后找出评价电影数量最多的用户和该用户的平均评分。
* **数据质量评估**：此步骤的重点是识别和处理丢失的数据、重复的记录以及纠正错误。此外，该过程旨在纠正这些问题，以确保后续分析产生准确可靠的结果。

为了保证数据集质量，我们反复地观察数据集地完整性，并使用填零、取平均值或者删除数据的方法来处理。

最终得到movie-rating.csv的数据集并保存，为了之后的神经网络模型搭建使用。

### 获取电影属性的额外信息

* 调用([The Movie DB](https://developer.themoviedb.org/docs/getting-started)) API

除了csv数据集中处理得到的信息以外，我们还把links.csv中的tmdbId调用了API来获取额外的信息。不过我们的数据集比较大，所以我们使用了python的 concurrent.futures.ThreadPoolExecutor 来加快地调用API。

我们从API选了一些额外信息，其中有：vote\_average, vote\_count, popularity, revenue, budget, runtime，添加到我们的数据集。

* 数据集处理

A screenshot of a computer

Description automatically generated我们再次对数据集进行处理，保证我们数据的质量。保存数据之前，我们对我们的添加额外信息后的数据集进行检查，检查每个特征与我们的目标(‘rating’)的相关性，避免无关的信息。最终保存为movie-rating-additional.csv，为之后的神经网络作为要处理的对象

以上是一个相关性矩阵，它显示了数据集中各变量之间的相关系数。相关系数是一种统计度量，用于计算两个变量的相对运动之间的关系强度。值的范围在-1.0和1.0之间。-1.0的相关性显示出完全负相关，而1.0的相关性显示出完全正相关。0.0的相关性显示两个变量之间没有线性关系。

一些相关性的简要解释：

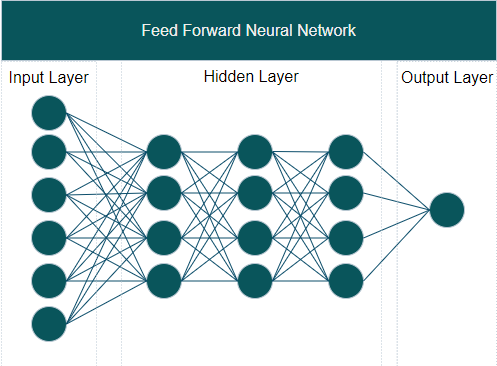
* movieId与目标变量rating的相关性为0.029186。这是一个非常弱的正关系，表明这两个变量之间几乎没有线性关系。
* Adventure与rating的相关性为-0.019107。这是一个非常弱的负关系，表明冒险电影的评分可能会略低，但关系不强。
* Animation与rating的相关性为0.069290。这是一个弱的正关系，表明动画电影的评分可能会略高。
* Drama与rating的相关性为0.158548。这是一个弱的正关系，表明剧情电影的评分可能会更高。
* vote\_average与rating的相关性为0.379752。这是一个中等的正关系，表明评分平均较高的电影的评分可能会更高。
* rating与自身的相关性为1.000000，这总是如此。

## 神经网路模型的设计

### 数据预处理:

使用panda库，引入我们以上分析并处理好的数据集后（包括额外信息的数据集）。我们还需要再一次对数据进行处理。因为每个数据的类型、大小不同，所以我们对此数据集进行Feature Scaling,数据类型转换等技术，使我们的数据可以在我们搭建的神经模型使用，作为Input Layer的数据。

### 模型的设计:

我们选的神经网络是前馈神经网络，如图：

我们设计的前馈神经网络（Feed-forward Neural Network），它包含三个线性层和两个ReLU激活函数。以下是网络设计的详细描述：

* 输入层：网络接收一个大小为input\_size的输入向量。
* 第一层：输入向量首先通过一个线性层，该层有256个神经元。然后，通过ReLU激活函数进行非线性变换。
* 第二层：第一层的输出再次通过一个线性层，该层也有256个神经元。然后，再次通过ReLU激活函数进行非线性变换。
* 输出层：最后，第二层的输出通过第三个线性层，该层有output\_size个神经元，生成网络的最终输出。

这个网络设计的目标是通过学习输入向量和输出之间的复杂映射关系，来进行预测任务。

### 模型的编译

这涉及Loss Function和Optimizer。损失函数用于计算模型在训练期间应寻求最小化的数量。优化器是用于改变神经网络属性以降低损失率的算法。

1. Loss Function

我们选用了MSELoss函数作为我们的Loss函数。

我们选择MSELoss函数的原因是：

* 惩罚大误差：MSE损失函数会对误差进行平方处理，这意味着大的误差会被更严重地惩罚。这在我们希望模型对所有预测都尽可能准确的情况下非常有用。
* 连续值处理：MSE损失函数可以处理连续值，这使得它非常适合用于回归问题。
* 模型性能度量：MSE损失函数衡量的是模型预测值与实际值之间的平均平方误差，因此它可以帮助我们理解模型在整个数据集上的性能3。
* 易于计算：MSE损失函数的计算过程简单，易于实现，这也是它受欢迎的一个原因。

1. Optimizer

我们选用了Adam Optimizer作为我们的优化器因为：

* 计算效率：Adam优化器在计算上非常高效，对内存需求较小，非常适合处理大规模的数据或参数。
* 自适应学习率：Adam优化器的一个关键特性是它能够为每个参数单独调整学习率，这使得它能够在复杂的损失函数中有效地进行优化和收敛。
* 偏差修正：Adam优化器在早期的训练阶段引入了偏差修正机制，有助于加速收敛。
* 易于实现：Adam优化器的实现相对直接，通常不需要大量的超参数调整。
* 结合了动态学习率和平滑处理：Adam优化器可以看作是RMSprop和带动量的随机梯度下降的结合。它像RMSprop一样使用平方梯度来调整学习率，同时像带动量的随机梯度下降一样使用梯度的移动平均而不是梯度本身

### 训练循环

我们把数据集随机分成90%的训练集和10%的测试集。

训练循环如下：

1. 将划分的训练集输入模型，得到输出
2. 计算模型输出和真实输出之间的损失
3. 进行反向传播之前，现将模型的梯度清零
4. 计算损失函数关于模型参数的梯度
5. 根据计算得到的梯度更新模型的参数。

经过反复的训练若干个epochs\_num之后，我们将会得到一个已训练好的模型。

### 模型评估

训练后，模型的性能在单独的验证数据集上进行评估。这有助于确保模型不会过度拟合训练数据，并且可以很好地推广到新的、未见过的数据。

我们选择了几个最著名模型评估方法，来对我们的模型进行评估：

* r2 --- 计算R²分数，这是一个统计量，表示由回归模型解释的因变量的方差的比例。
* mse --- 计算均方误差（MSE），这是实际值和预测值之间的平方差的平均值。
* rmse --- 计算均方根误差（RMSE），这是MSE的平方根，提供了预测误差的直观度量。
* mae --- 计算平均绝对误差（MAE），这是实际值和预测值之间的绝对差的平均值。

### 调优模型参数:

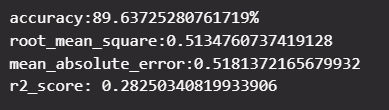
我们反复修改一些关键的参数以优化模型的性能：

* epochs\_num：这是训练的总轮数。每一轮（epoch）都会遍历整个训练数据集一次。
* learning\_parameters（lr）：这可能指的是学习率，它决定了模型参数在每次更新时的调整幅度。
* activation\_function：这是神经网络中的激活函数，如ReLU、sigmoid或tanh。激活函数可以引入非线性，使得神经网络能够学习更复杂的模式。
* network\_layer：增加神经网络层的数量或每层的神经元数量可以增加模型的复杂度。
* 通过调整这些参数，我们可以找到最适合特定任务的模型配置。这个过程通常需要多次尝试和验证，因为不同的任务可能需要不同的参数设置。

最终我们得到效果最好结果的模型参数是:

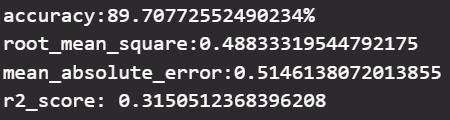
* epochs\_num = 100
* lr = 0.011
* activation\_function = ReLU()
* network\_layer = 3 层 256个神经元的线性层

# 模型的评估

第一次的测试得到的准确率

A black background with white numbers

Description automatically generated第二次的测试得到的准确率：

第三次的测试得到的准确率:

模型的平均评估：

* 准确率（Accuracy）：89.58%。这表示模型在所有预测中有大约89.58%是正确的。
* 均方根误差（Root Mean Square Error，RMSE）：0.50。这是预测值与实际值之间差异的平方和的平均值的平方根。RMSE越小，说明模型的预测性能越好。
* 平均绝对误差（Mean Absolute Error，MAE）：0.51。这是预测值与实际值之间差异的绝对值的平均值。MAE越小，说明模型的预测性能越好。
* 决定系数（R^2 Score）：0.30。这是模型解释目标变量变异性的能力。R^2 Score的值在0到1之间，越接近1，说明模型的预测性能越好。

综上所述，该模型的回归精度达到了 89.58%，这表明该模型可以很好地预测电影星级的评估,不过模型还是存在一些改进空间以获取更好的效果。

# 实验总结与思想

## 实验总结

结果：该模型在验证集上取得了令人满意的性能，表明它能够很好地从训练数据中进行泛化。该模型的预测与实际收视率密切相关，证明了其根据给定属性预测电影收视率的有效性。

结论：该实验证明了使用回归模型根据电影属性预测电影星级评分的可行性。未来的工作可以探索使用更复杂的模型或包含其他功能，例如电影评论或社交媒体情绪。

## 实验思想

Python 机器学习课程中的实验任务提供了一种理解复杂概念的实用方法。使用回归分析来预测电影星级预测等任务使我能够将理论知识应用到现实世界中。看到不同的特征如何影响模型的预测以及数据预处理和特征工程如何在提高模型性能方面发挥关键作用，真是令人着迷。

Python 机器学习课程是一次全面且富有启发性的旅程。它为 Python 编程和机器学习原理奠定了坚实的基础。该课程强调实际作业和现实世界的应用，使学习过程变得有吸引力和相关性。讲师知识渊博且平易近人，即使是最复杂的主题也能被理解

本课程激发了学生们对机器学习及其应用的浓厚兴趣。我希望进一步加深自己在该领域的知识，探索深度学习和强化学习等更高级的主题。我也对机器学习在解决现实问题方面的潜力感到兴奋，并希望以有意义的方式应用我的技能，也许在医疗保健或环境保护等领域。