# 《开源软件基础》课程大作业

|  |
| --- |
| 针对Scikit-learn仓库的分析和应用 |

题目：

|  |  |  |  |
| --- | --- | --- | --- |
| **姓名** | **学号** | **班级** | **成绩** |
| **李忠赫** | **20222241199** | **软件2208** |  |
| **程泽** | **20222241282** | **软件2208** |  |
| **莫竣凯** | **20222241388** | **软件2208** |  |
| **闵子洋** | **20222241280** | **软件2207** |  |
| **王紫怡** | **20222241400** | **软件2207** |  |

大连理工大学软件学院

2025年01月

**目录**

一、课题背景……………………………………………………………3

二、实验分析……………………………………………………………4

2.1 sklearn仓库结构………………………………………………4

2.2仓库近期历史提交信息分析……………………………………6

2.3 sklearn官网示例分析……………………………………… 10

三、实验设计……………………………………………………………13

3.1 具体实验方向…………………………………………………13

3.2 数据集预处理…………………………………………………13

3.3 可视化出租车费用流量………………………………………15

3.4 创建适用于循环神经网络的规范数据集……………………16

3.5 循环神经网络模型定义………………………………………17

3.6 训练模型………………………………………………………19

3.7 模型预测结果可视化…………………………………………19

四、实验结论……………………………………………………………21

1. 课题背景

scikit-learn(简称sklearn)是机器学习领域中最受欢迎的Python库之一。该库的开发始于2007年，当时是David Cournapeau的Google Summer of Code项目。那年晚些时候，Matthieu Brucher开始研究这个项目，并作为他论文的一部分。

2010年，Fabian Pedregosa、Gael Varoquaux、Alexandre Gramfort和Vincent Michel领导了INRIA项目，并于2010年2月1日发布了第一个公开版本。从那时起，大约3个月的周期出现了几个版本，蓬勃发展的国际社区一直在领导开发工作。

21 世纪以来，机器学习领域快速发展，新算法不断涌现，如支持向量机、随机森林、梯度提升等。不同算法适用于不同场景，开发者需一个集中平台，方便获取和使用各类算法。例如，在处理高维数据分类时，支持向量机效果良好；而处理大规模数据回归问题，随机森林可能更合适。sklearn将这些算法整合，为开发者提供统一接口。除此之外，机器学习应用领域不断拓展，涵盖计算机视觉、自然语言处理、生物信息学等，由于各领域对机器学习工具需求不同，sklearn旨在满足多样化需求，提供通用且灵活的机器学习框架。比如在图像分类中，可利用sklearn的分类算法对图像特征进行分类；在自然语言处理中，用于文本分类和聚类。

本课题将对sklearn仓库进行深入探究，着重分析sklearn仓库的历史提交信息，并从多种机器学习领域利用sklearn库进行实操探究，主要分析sklearn库和循环神经网络相结合的操作可行性。我们的代码详细可见于：<https://github.com/Lizhonghe0214/Scikit-learn_Analysis>。

1. 实验分析

2.1 sklearn仓库结构

Scikit-learn的官方代码库结构组织有序，涵盖项目配置、基准测试、工具、文档、示例和核心代码等多个关键部分，为项目的开发、维护和使用提供便利。

项目配置包含.binder文件夹，.circleci文件夹，.github文件夹和.gitattributes、.gitignore等与版本控制相关文件。

1. .binder文件夹：主要用于配置Jupyter Binder环境。Jupyter Binder能将代码仓库一键转化为可交互的在线计算环境，用户无需本地安装scikit-learn及其复杂的依赖，直接在网页上就能运行代码示例、测试新想法，方便对库进行探索和学习，促进项目的传播和使用。
2. .circleci文件夹：存放与CircleCI持续集成相关的配置文件。持续集成能在每次代码更新时自动执行构建、测试等操作，确保代码质量。该文件夹内的配置文件规定了在CircleCI平台上的构建环境、测试命令、依赖安装等流程，帮助开发者及时发现代码合并中的问题。
3. .github文件夹：包含Github相关的配置和工作流程文件。

基准测试包含asv\_benchmarks文件夹和benchmarks文件夹，用于存放基准测试相关代码和数据。通过这些基准测试，能评估 scikit-learn 中不同算法和功能的性能表现，帮助开发者了解代码优化方向，对比不同版本的性能差异，判断新功能或修改是否影响整体性能，为优化代码和算法提供依据。

build\_tools文件夹和maint\_tools文件夹用于工具脚本。build\_tools文件夹包含构建项目的工具脚本，如生成安装包、编译扩展模块等；maint\_tools 文件夹存放项目维护的工具脚本，涵盖代码检查、代码格式化、数据预处理等日常维护任务相关代码，提升开发效率和代码规范性。

doc文件夹用于存放项目文档源文件，最终生成HTML格式的官方文档。包括用户指南、API参考、教程等内容，帮助用户快速上手和深入理解scikit-learn。

examples文件夹内提供丰富的示例代码，展示scikit-learn在各类机器学习任务中的具体应用。如分类、回归、聚类等，新手可通过学习这些示例快速掌握库的使用方法。

sklearn文件夹内包含scikit-learn库的核心代码，实现了众多机器学习算法和工具。包含分类算法（如逻辑回归、支持向量机）、回归算法（如线性回归、岭回归）、聚类算法（如K-Means聚类）等算法和数据预处理、模型评估等工具，是库的功能实现主体。

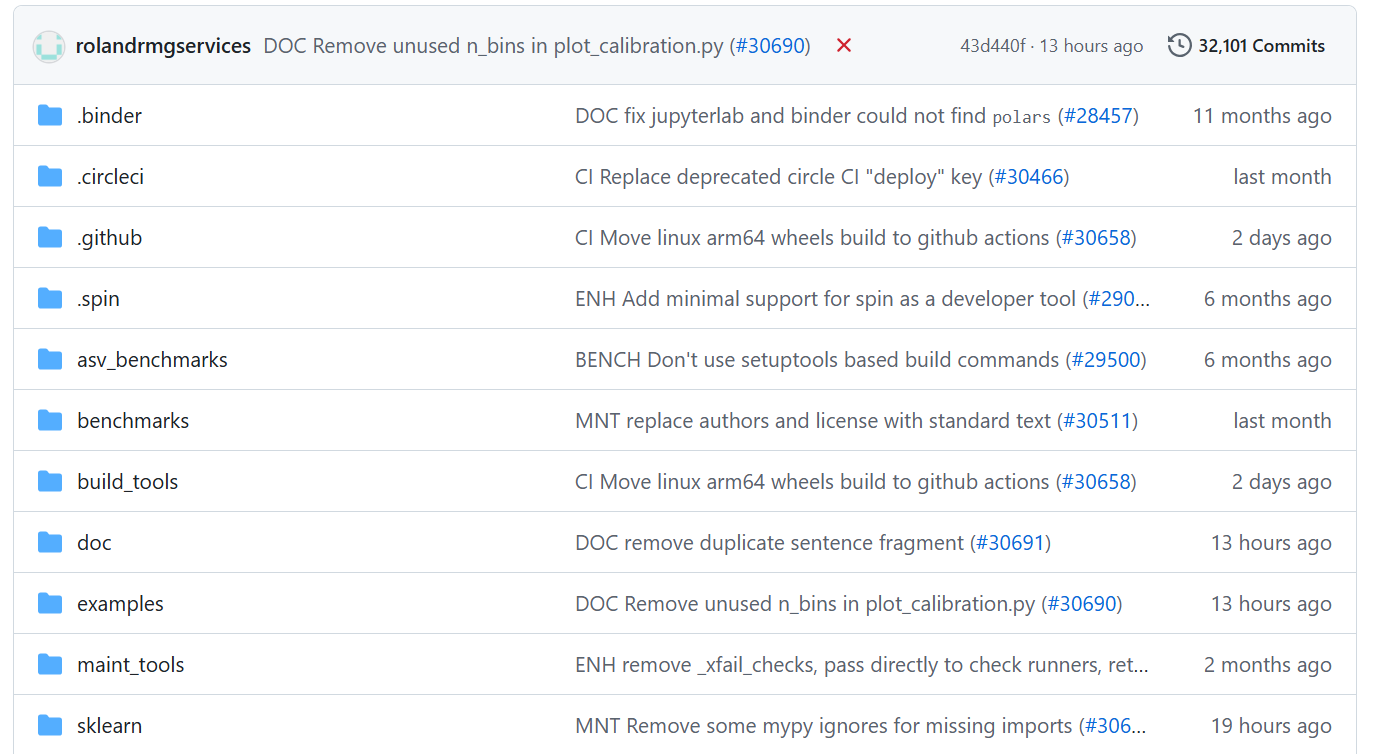


图 1 scikit-learn仓库Code主页文件夹一览图

除了文件夹，仓库中的这些文件在项目的开源许可、构建安装、代码质量管理、功能测试以及用户使用引导等方面发挥着关键作用，确保项目顺利运行和推广。

1. README.rst：项目的说明文档，涵盖项目概述（是基于Python的机器学习模块，基于SciPy开发且遵循BSD-3-Clause许可）、安装方法（对依赖的Python、Numpy等版本要求，及pip、conda安装指令）、开发相关信息和重要链接（官网、代码库、下载地址等），是了解项目的入口。
2. CODE\_OF\_CONDUCT.md：行为准则文件，规范社区成员参与项目开发、交流等活动的行为，营造积极、包容、协作的社区环境，保障项目社区健康发展。
3. CONTRIBUTING.md：指导开发者如何为项目做出贡献，包括代码贡献的流程、规范、测试要求等，帮助新贡献者快速了解如何参与到项目的开发和维护中。
4. SECURITY.md：记录项目的安全相关信息，例如如何报告安全漏洞、项目的安全策略等，帮助开发者和用户保障项目的安全性。
5. COPYING：项目的开源许可证文件，明确项目代码的使用、分发和修改权限。
6. Makefile：包含一系列规则和命令，用于自动化构建、测试、安装等任务。
7. .mailmap：用于将多个Git作者的名称和邮箱映射到规范的名称和邮箱，在处理多人协作且作者信息可能不规范的情况下，帮助统一作者信息，使提交历史更加清晰。
8. .pre-commit-config.yaml：配置pre-commit钩子，在代码提交前自动执行一些检查和格式化任务，如代码风格检查、代码格式化、安全性检查等，确保提交的代码符合项目规范。
9. CITATION.cff：提供项目的引用信息，方便科研人员在学术论文等场景中正确引用scikit-learn项目。

除此之外，.currus.yml、.codecov.yml、.coveragerc、azure-pipelines.yml等为针对不同工具、环境的配置文件。



图 2 scikit-learn仓库Code主页文件一览图

2.2 仓库近期历史提交信息分析

我们聚焦于scikit-learn仓库从2025年1月16日到1月22日这七天的历史提交信息，scikit-learn仓库十分活跃，提交信息数量可观且种类繁多，涵盖文档完善、测试优化、代码修复、功能增强和CI配置更新等方面。这从多维度推动项目发展，确保其稳定性、易用性和功能性不断提升。

接下来我们将从1月22日倒序梳理提交信息并分析。

1月22日有四份提交，分别属于文档优化（#30576、#30696）、测试（#30517）和项目维护（#30697）范畴。

DOC impact of stratification on the target class in cross-validation splitters(#30576)旨在阐述交叉验证分割器中分层对目标类的影响，完善该部分文档，能帮助开发者更精准把握交叉验证机制，从而优化模型评估策略，合理选择和调整模型参数。 DOC Move legend to avoid hiding data points(#30696)移动图例避免遮挡数据点，改善了可视化效果。TST Use global\_random\_seed in sklearn/cluster/tests/test\_mean\_shift.py(#30517)增强测试的可重复性。MNT Fix binder notebook generation(#30697) 修复 Binder notebook 生成问题，保障用户能借助 Binder 在网页端便捷运行 notebook。

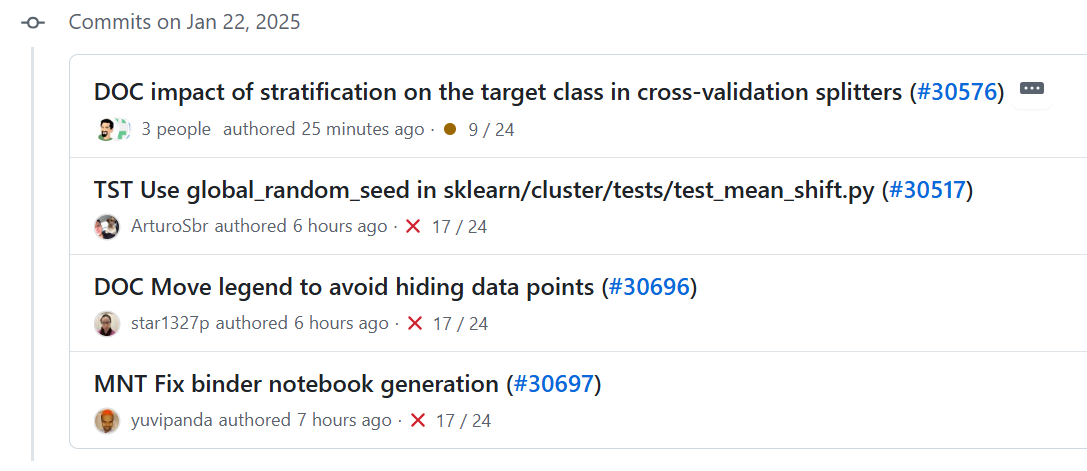


图 3 sklearn仓库1月22日提交信息

1月21日有6份提交，分别属于文档优化（#30690、#30691）、项目维护（#30687、#30671、#30685）和代码修复（#30649）范畴。

DOC Remove unused n\_bins in plot\_calibration.py(#30690)进行文档清理，清除冗余信息，使文档更简洁明了。DOC remove duplicate sentence fragment(#30691)去除重复句子片段，优化了文档表述。MNT Revert robots.txt addition(#30687)撤销了之前添加的robots.txt的操作，合理管理网站索引。MNT Remove some mypy ignores for missing imports(#30671) 清理部分针对缺失导入的 mypy 忽略，可增强代码类型检查的严格性。MNT Add robots.txt to avoid indexing of old version doc(#30685) 添加robots.txt防止旧版本文档被索引，可避免用户获取到过时信息，保证用户接触到的文档是最新且准确的，提升用户获取信息的准确性和有效性。FIX Validate estimators in Voting{Classifier,Regressor}(#30649) 在投票分类器和回归器中添加对估计器的验证，能确保输入的估计器有效，增强模型运行的稳定性和可靠性。

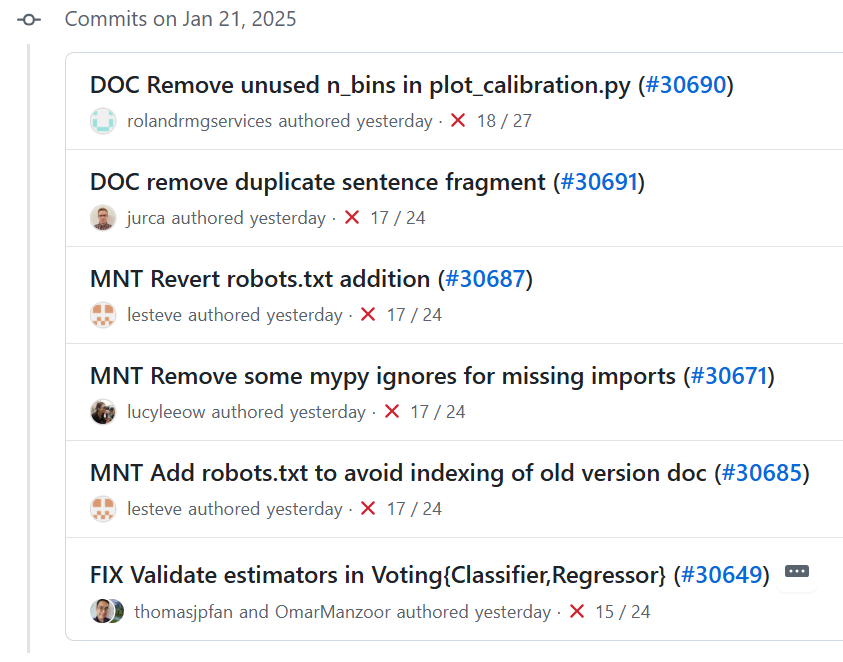


图 4 sklearn仓库1月21日提交信息

1月20日有9份提交，涵盖文档优化（#30665）、代码修复（#30483、#30668）、持续集成（#30681、#30658、#30679、#30678、#30677）和项目维护（#30606）。

DOC make the graph more readable(#30665)优化图表可读性。FIX Restore support for n\_samples==n\_features in MinCovDet(#30483)恢复MinCovDet对n\_samples==n\_features情况的支持，拓宽了该算法的适用范围。FIX update deprecated param for example using class\_likelihood\_ratios(#30668)更新使用示例中已弃用的参数，让用户在参考示例时能顺利实践。CI Xfail test for Pyodide(#30681)针对Pyodide环境标记测试为预期失败。CI Move linux arm64 wheels build to github actions(#30658)将Linux ARM64架构的轮子构建任务迁移到GitHub Actions，可优化构建流程，并借助GitHub Actions的优势。CI Update lock files for array-api CI build(s)(#30679)、CI Update lock files for free-threaded CI build(s)(#30678)和CI Update lock files for cirrus-arm CI build(s)(#30677)分别更新了array-api CI、free-threaded CI和cirrus-arm CI构建的锁文件，保证构建环境的稳定性和一致性。MNT Update conda-lock version to 2.5.7(#30606)更新了conda-lock的版本。



图 5 sklearn仓库1月20日提交信息

1月17日有5份提交，涵盖功能增强（#29288）、持续集成（#30661、#30496）、性能优化（#30415）和代码修复（#30644）。

ENH Add replace\_undefined\_by param to class\_likelihood \_ratios(#29288)为函数添加新参数，增加了功能灵活性。CI Update lock-file to fix broken pip on main(#30661)更新锁文件，修复main分支上虽坏的pip问题。CI Use scipy-doctest for more convenient doctests(#30496)使用scipy-doctest进行文档测试。PERF float32 propagation in GaussianMixture (#30415)优化GaussianMixture中float32的传播，可提高该算法的性能。FIX Sample weight in BayesianRidge(#30644)修复BayesianRidge中的样本权重问题，提升模型的准确性和可靠性。



图 6 sklearn仓库1月17日提交信息

1月16日有3份提交，分别属于功能增强工作（#30155）、文档修复工作（#30654）和项目维护工作（#30651）。

ENH add sample weight support to MLP(#30155)为多层感知器（MLP）添加样本权重支持，使MLP在处理数据时能考虑样本权重，提升模型的适应性，拓宽了 MLP 的应用场景。DOC Fix link in contributing.rst(#30654)修复链接问题，保证开发者在查看贡献指南时，链接能正常跳转。MNT Tweak towncrier fragments to follow guideline(#30651) 调整towncrier片段以遵循指南，有助于规范项目版本发布时的变更日志生成。



图 7 sklearn仓库1月16日提交信息

2.3 sklearn官网示例分析

scikit-learn在机器学习领域应用繁多，为此，我们首先登录sklearn官网 <https://scikit-learn.org，学习sklearn>库的使用方法。我们选取一典型机器学习模型决策树，并对其中一例子：绘制在iris数据集上训练的决策树的决策面，进行复现并学习sklearn库的使用技巧。

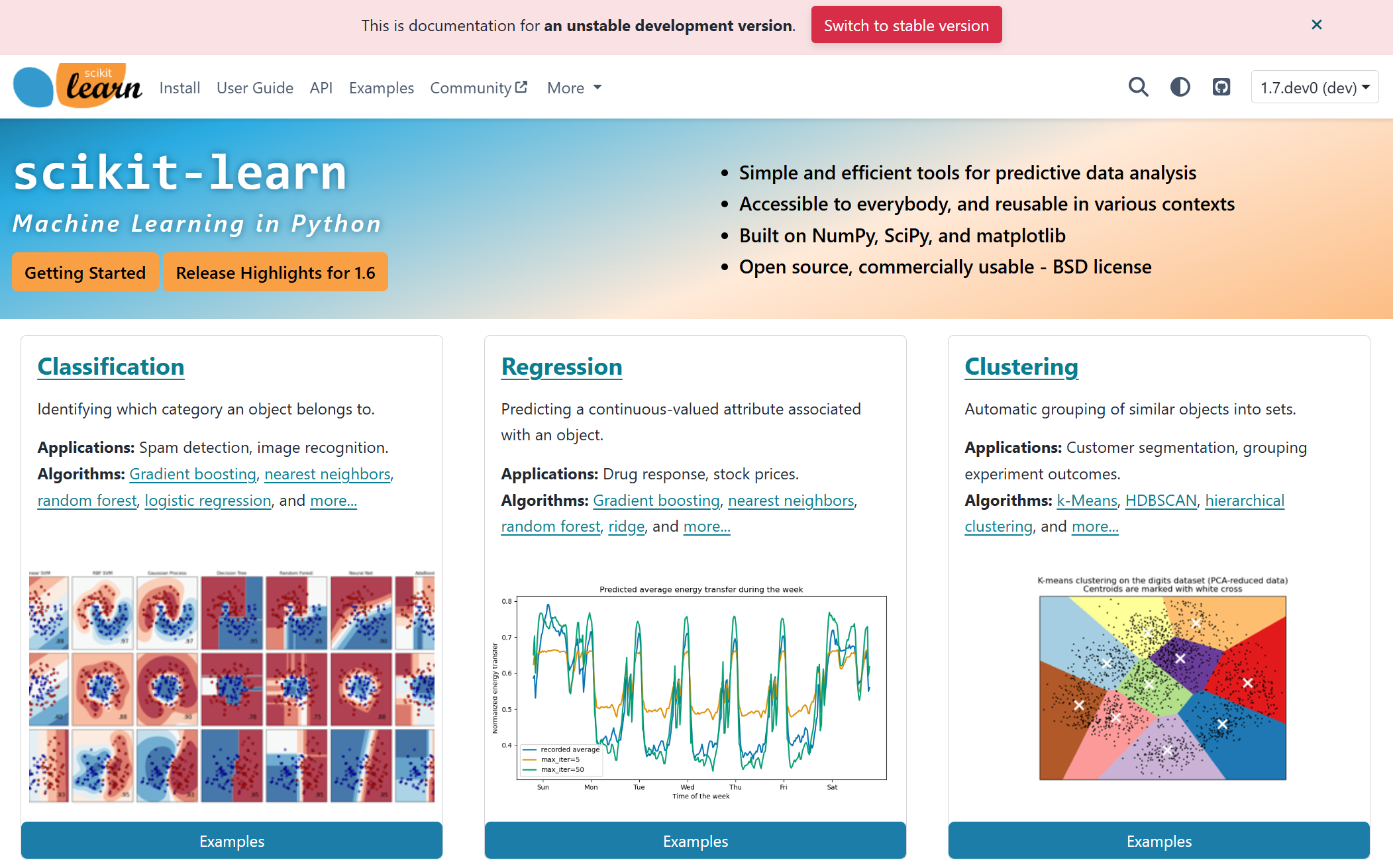


图 8 sklearn官网首页

首先导入必要的库，加载数据集并设置参数，iris数据集有3个类别，设置绘图时每个类别的颜色分别为红黄蓝，设置绘制决策边界的步长为0.02。

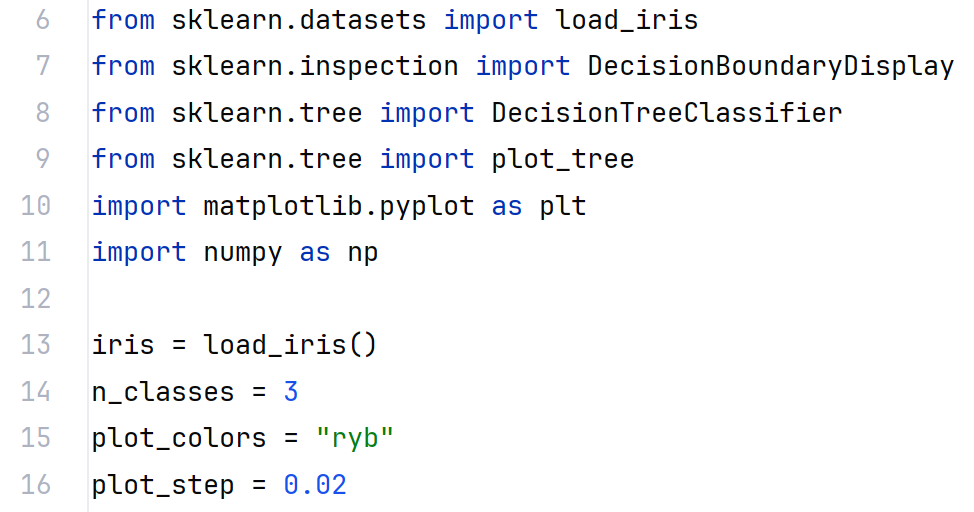


图 9必需的库和参数设置代码

下一步，显示在所有特征对上训练的树的决策函数。遍历不同的特征组合，每次取2个特征进行分析，训练好决策树分类器并绘制决策边界，之后接着绘制训练数据点。

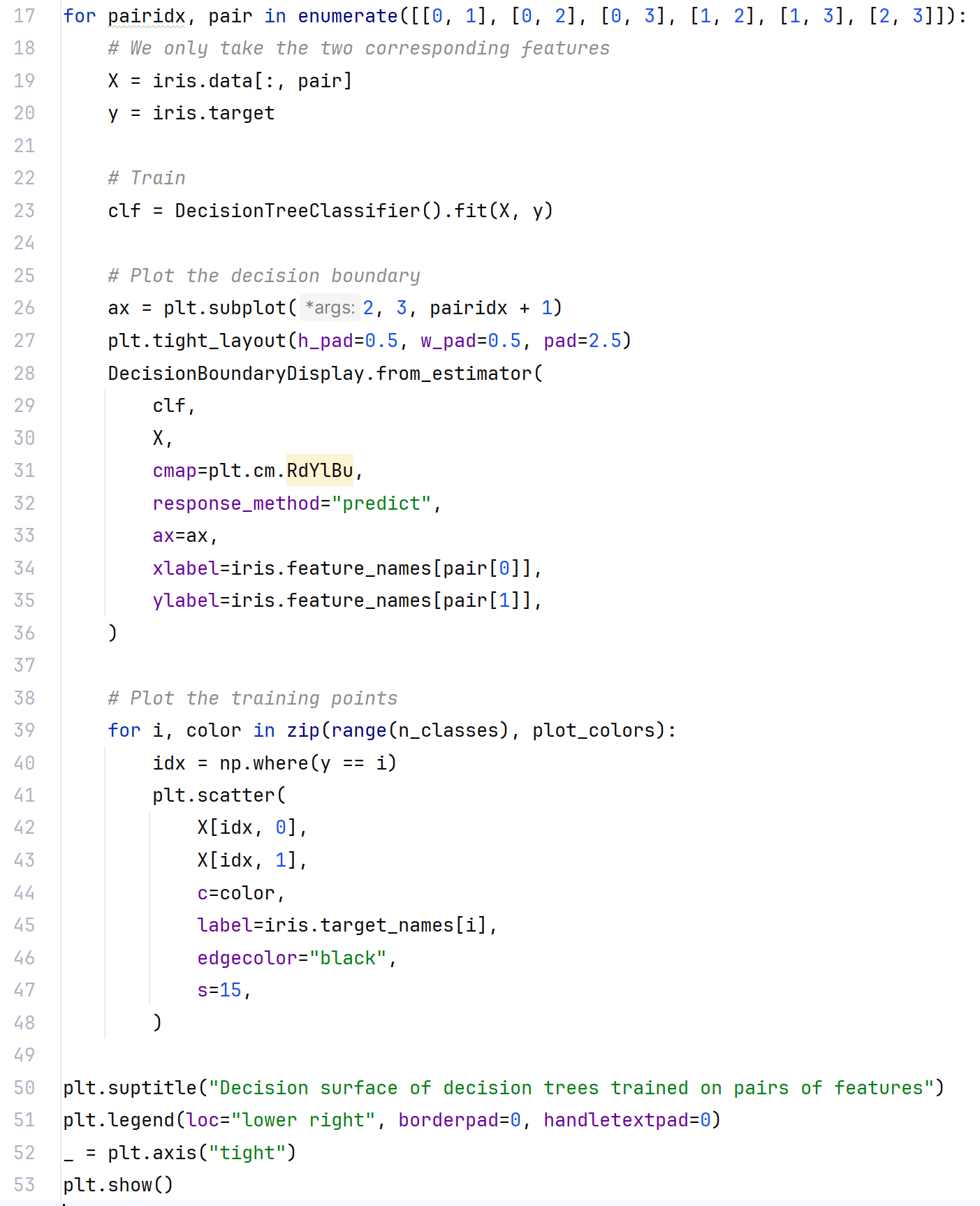


图 10决策树训练及绘制决策边界代码

决策边界如图11所示。



图 11 iris数据集的决策边界

最后显示在所有特征上同时训练的单个决策树的结构。

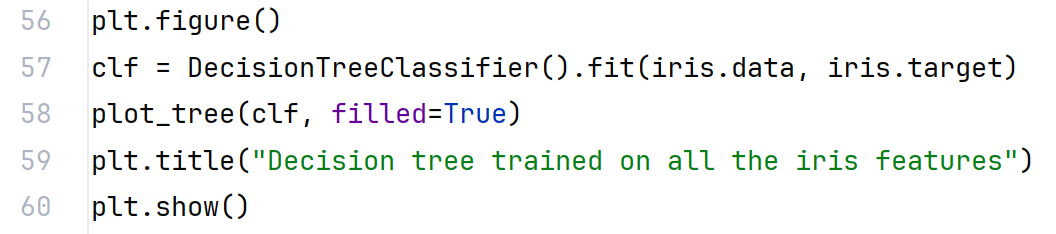


图 12所有特征上同时训练的单个决策树代码

结构如图13所示。

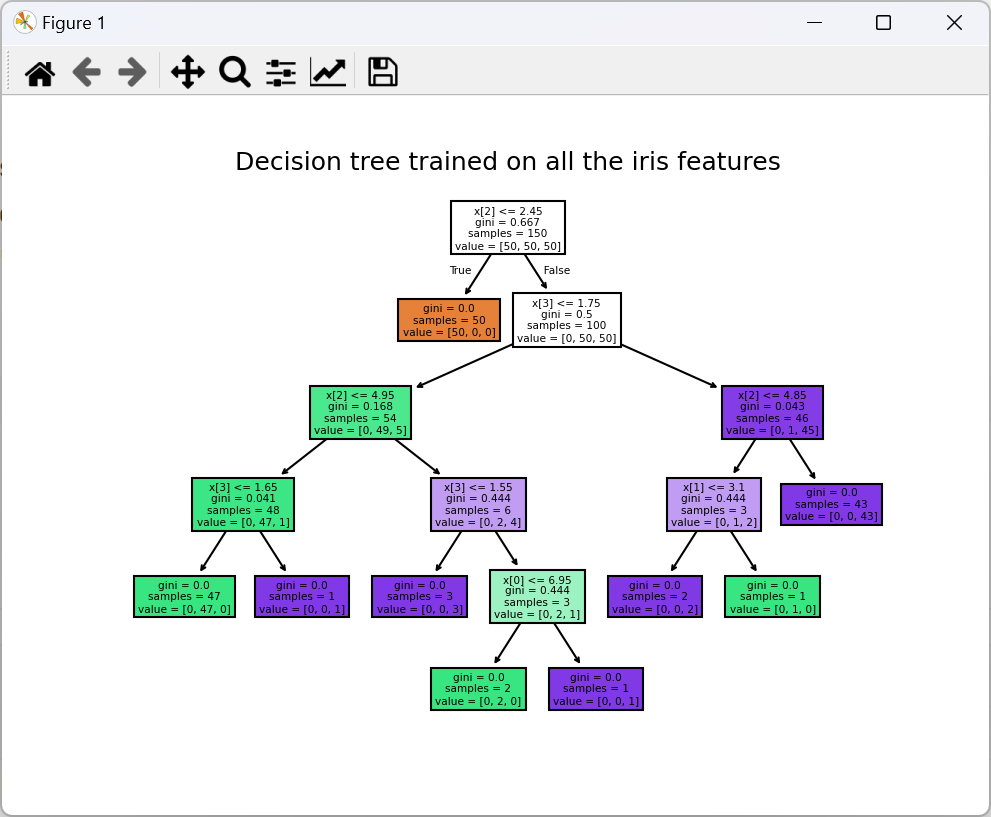


图 13决策树结构图

1. 实验设计

3.1 具体实验方向

现在，为深入探究scikit-learn的泛用性，拟采用pytorch框架，并以sklearn库为辅助，利用循环神经网络进行出租车费用流量预测，深入分析sklearn库的实用意义。

3.2 数据集预处理

纽约市绿牌出租车数据集特征众多，为了准确把握出租车流量特征，拟采用每小时的费用总和作为流量的表示。数据集特征信息可见于图14。

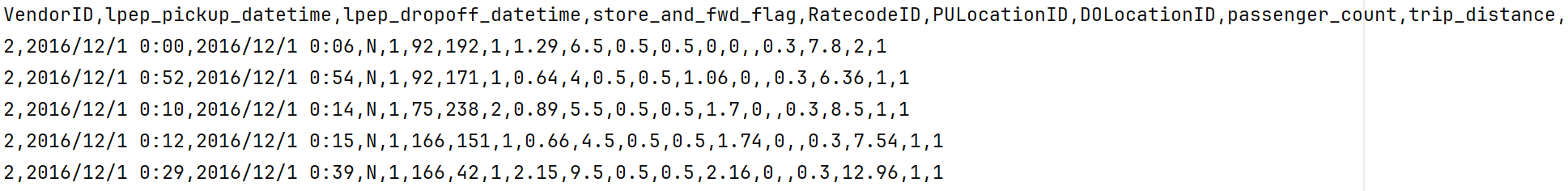


图 14纽约市绿牌出租车数据集

通过对数据集total\_amount项的总体观察，为了排除异常值对于数据集的干扰，预备数据清洗函数来剔除含异常total\_amount值的样本。设置保留的样本的total\_amount值均在0-100之内。

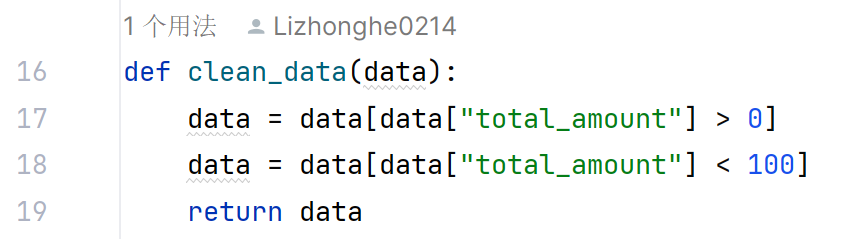


图 15数据清洗函数代码

首先导入数据集csv文件，取整前1000000行数据，筛选出需要的特征列：lpep\_pickup\_datetime和total\_amount。Lpep\_pickup\_datetime表示出发时间，total\_amount表示车费总金额。首先剔除缺失值，再调用预备函数剔除异常值。之后对数据集进行变形，将lpep\_pickup\_datetime列设置为索引，并按每六十分钟的数据进行聚合

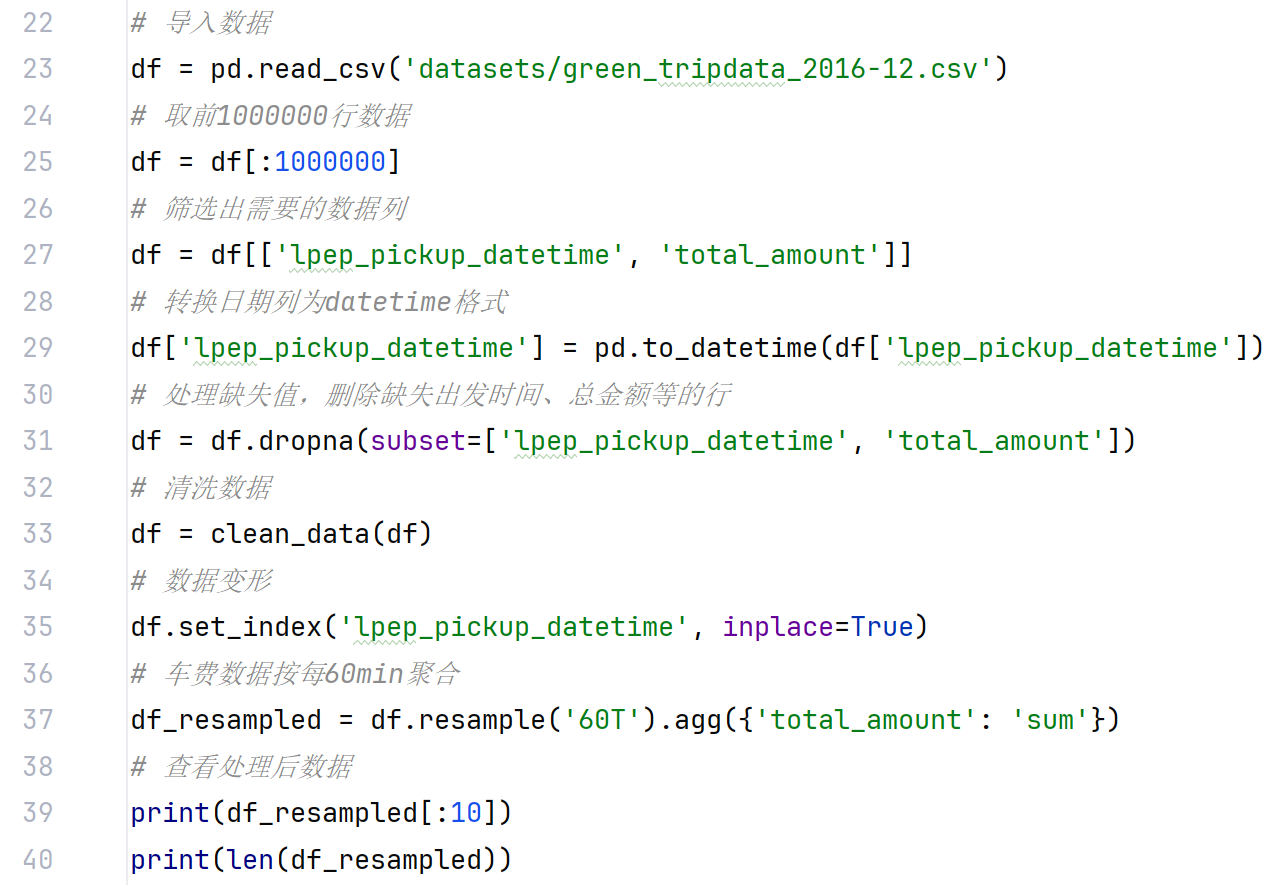


图16数据集预处理主要代码

预处理后的数据集的前10行数据及数据集长度如图17所示，现在数据集总长度为577。

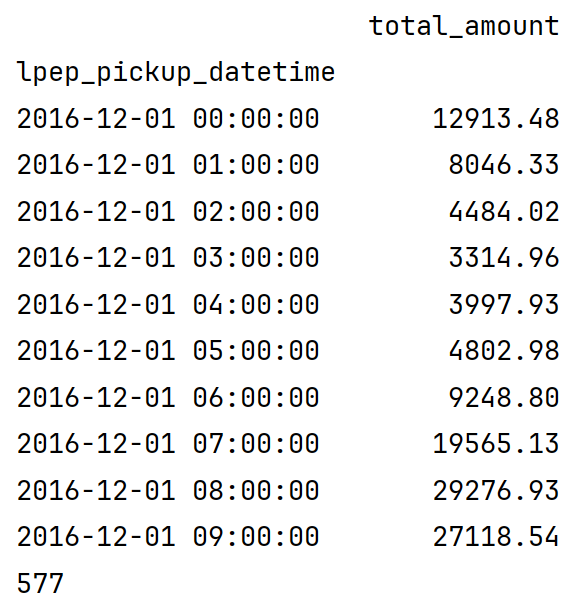


图 17预处理后数据集信息

接下来将数据集分离为输入和输出。输入X为数据集的索引，即时间戳（间隔为60min），输出y为数据集对应索引的值，即每小时的费用总和，间接代表每小时的出租车流量。

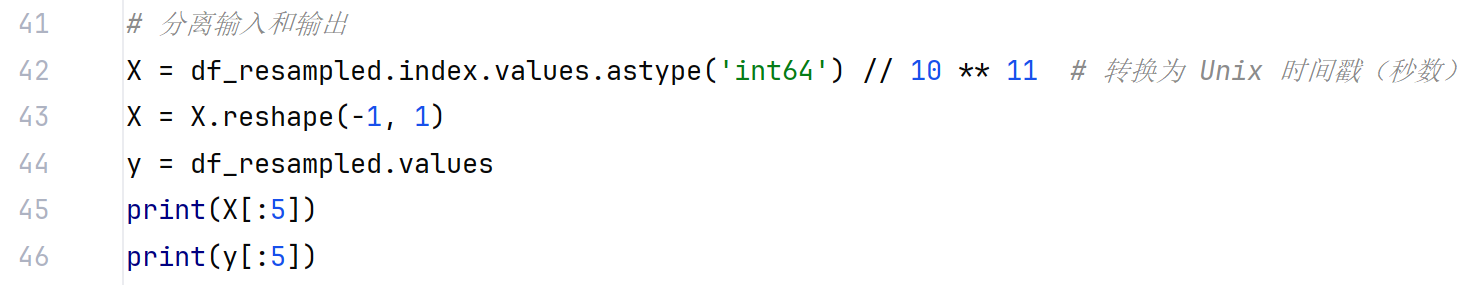


图 18分离输入和输出代码

3.3 可视化出租车费用流量

利用matplotlib库绘制每60min的纽约市绿牌出租车费用流量数据的折线图。

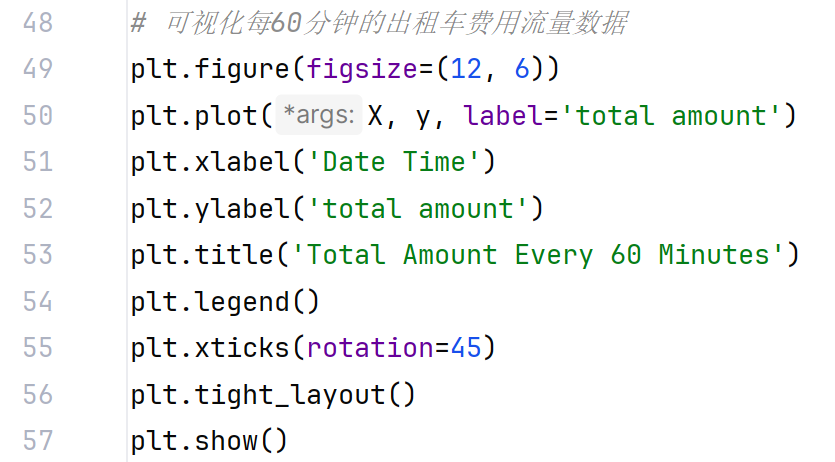


图 19 GR、出租车费用流量可视化代码

结果如图20所示。观察图像发现，曲线随着时间的增长呈现出周期性的不规则的振动变化，分析可得一个周期长度即为一天左右，这符合出租车流量的以天为单位的周期性规律。

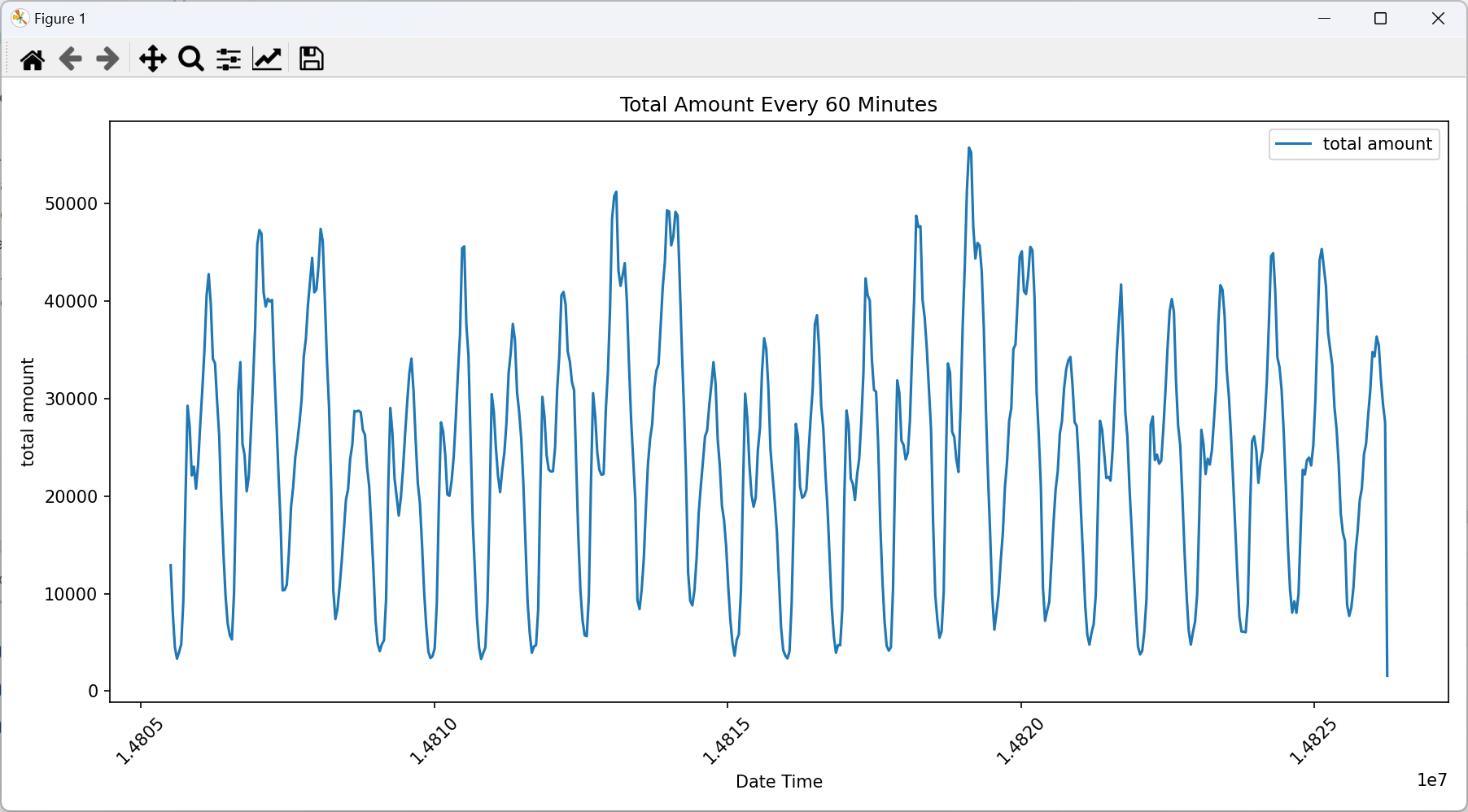


图 20 出租车费用流量折线图

3.4 创建适用于循环神经网络的规范数据集

为了创建规范的适用于循环神经网络模型的时间序列数据集，首先对数据进行规范化处理。第一步，对数据类型进行转换，将所有输出y的类型转换为float32（单精度浮点数），以此来满足后续的LSTM、RNN和GRU模型训练时对数据类型的要求。之后，调用scikit-learn库的MinMaxScaler类进行数据归一化操作，加快循环神经网络模型的拟合，降低出现过拟合的风险。

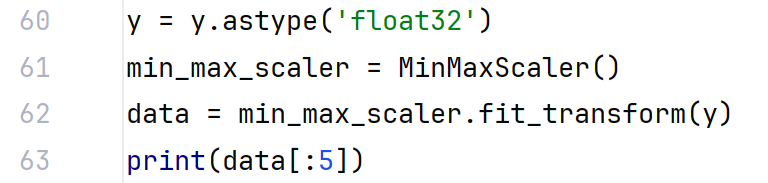


图 21数据规范化代码

定义创建数据集函数create\_dataset，参数dataset表示输入的一维时间序列数据，参数look\_back表示回看的历史时间步数，默认为10。函数的功能即为遍历时间序列数据，将截取的前look\_back个时间步的历史数据添加到data\_X中，将当前历史数据之后的下一时间步的数据添加到data\_y中，最后将data\_X和data\_y列表转换为numpy数组并返回。之后设置历史时间步数sequence\_length为24，由于采用的出租车费用的时间序列数据之间的间隔为60min，24个时间步即代表回看24小时内的数据，即利用前一天的纽约市绿牌出租车费用流量数量来预测下一小时的费用流量。

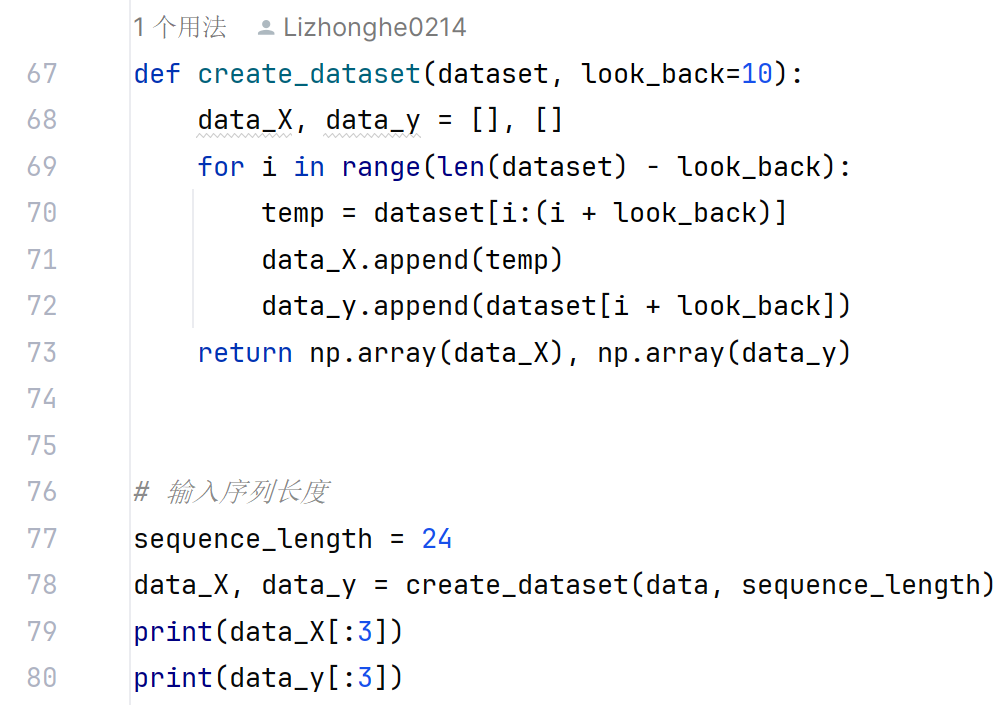


图 22创建数据集函数代码

接下来将数据集分割为训练集和验证集，其中前70%的数据为训练集，后30%的数据为测试集。由于采用pytorch框架，因此将形为numpy数组的寻来你输入数据train\_X转换为pytorch的张量，接着将张量的数据类型转换为单精度浮点数，最后利用view操作将train\_X转换为一个三维张量，形状为（训练样本数量，序列长度，特征维数），适合作为LSTM、RNN和GRU等序列模型的输入。利用相同的思想转换训练输出数据train\_y和测试输入数据test\_X。train\_y形状为（训练样本数量，特征维数），test\_X形状为（测试样本数量，序列长度，特征维数）。

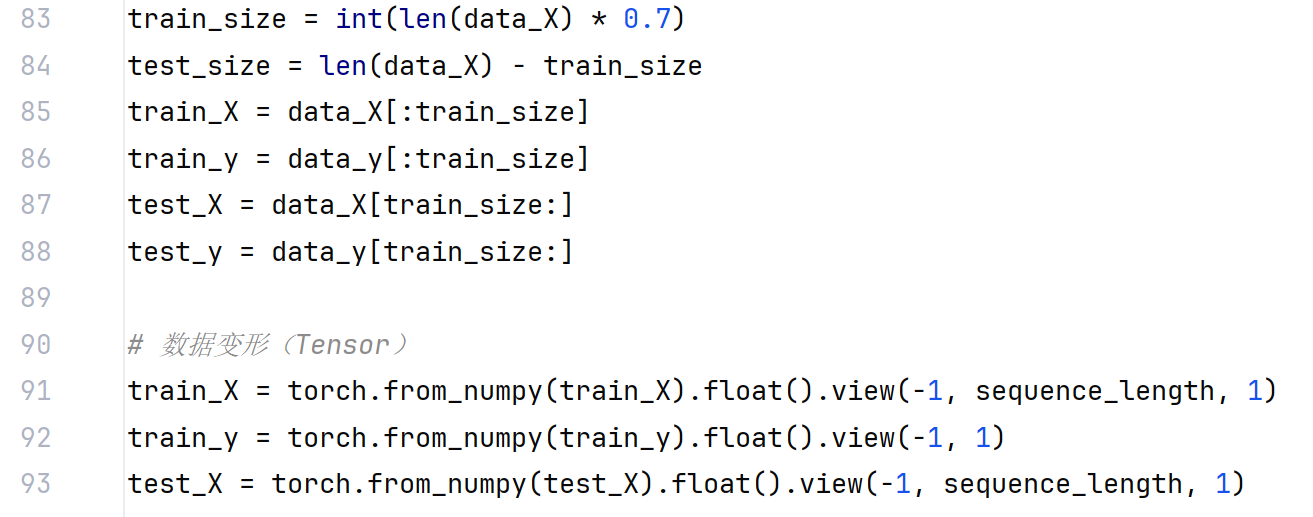


图 23 GR数据集分割并转换为张量代码

3.5 循环神经网络模型定义

为了探究不同循环神经网络模型在该纽约市绿牌出租车数据集上的训练效果，现定义LSTM、RNN和GRU三种模型，分别实现为SimpleLSTM类、SimpleRNN类和SimpleGRU类。为了控制变量，三种模型初始化时隐藏层大小均设置为64，层数均为2。在模型的初始化中定义循环神经网络层（参数batch\_first=true表示输入和输出张量的第一个维度是批量大小）和全连接层，在前向传播方法forward函数中定义输入数据x是如何在模型中流动并得到最终的输出的：先将x传入循环神经网络层进行计算，之后从该层的输出中提取每个样本的最后一个时间步的输出传入全连接层进行线性变换，得到最终的预测结果。

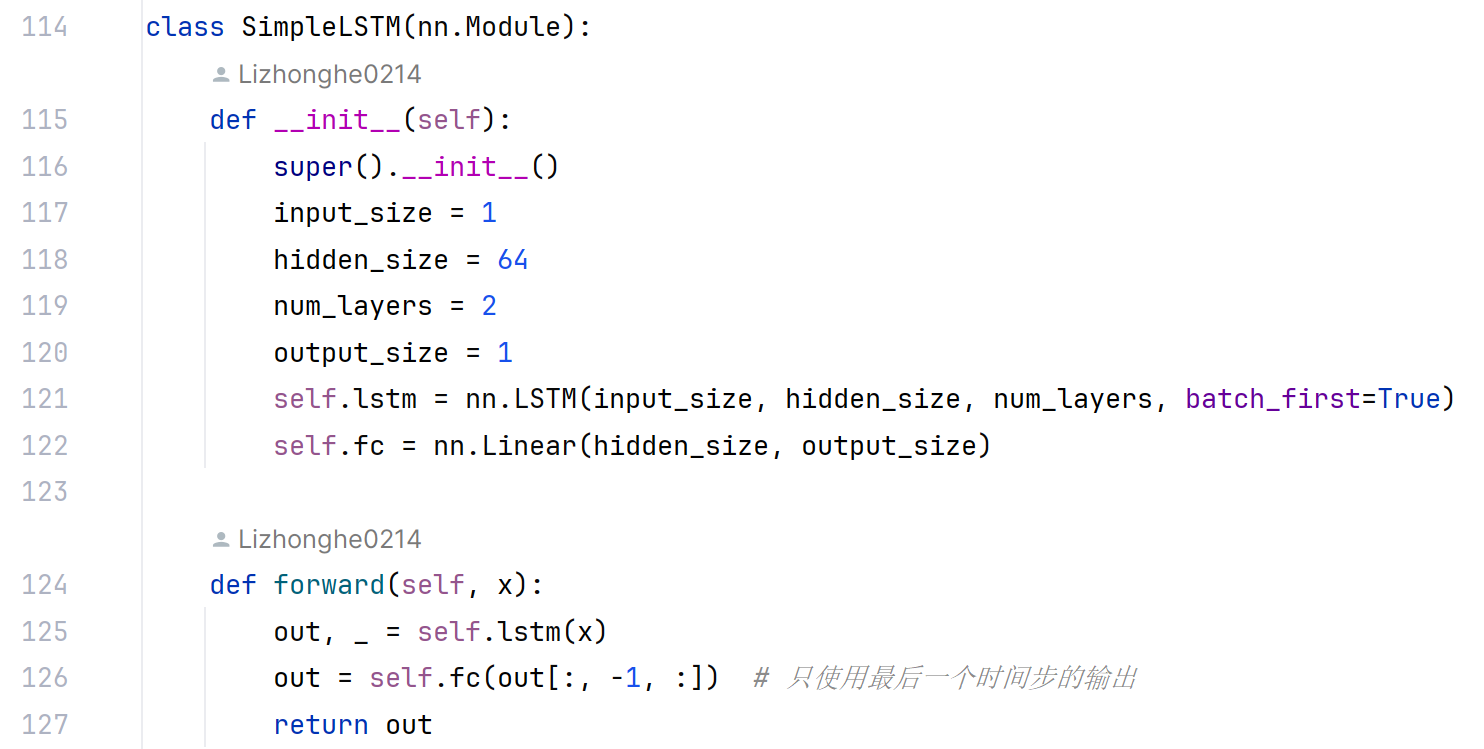


图 24 LSTM模型定义代码

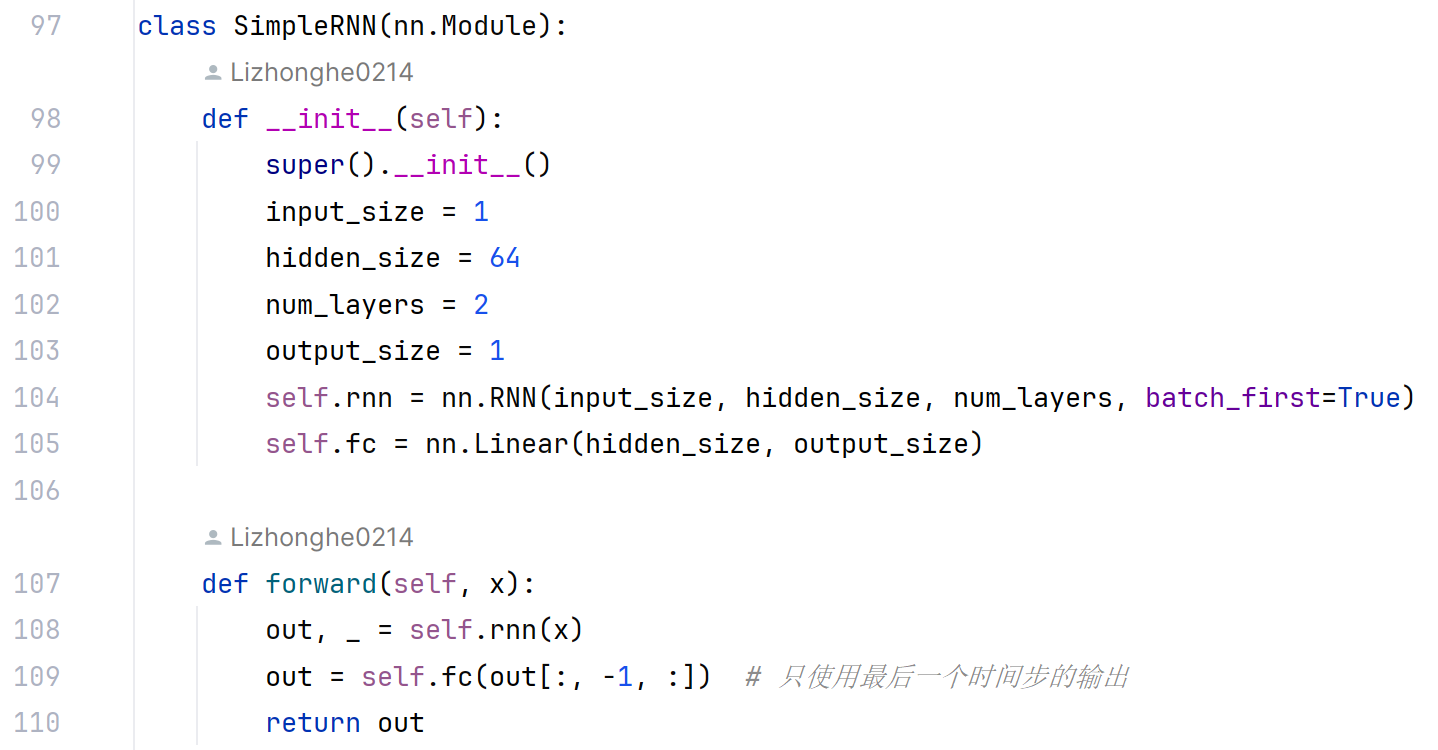


图 25 RNN模型定义代码

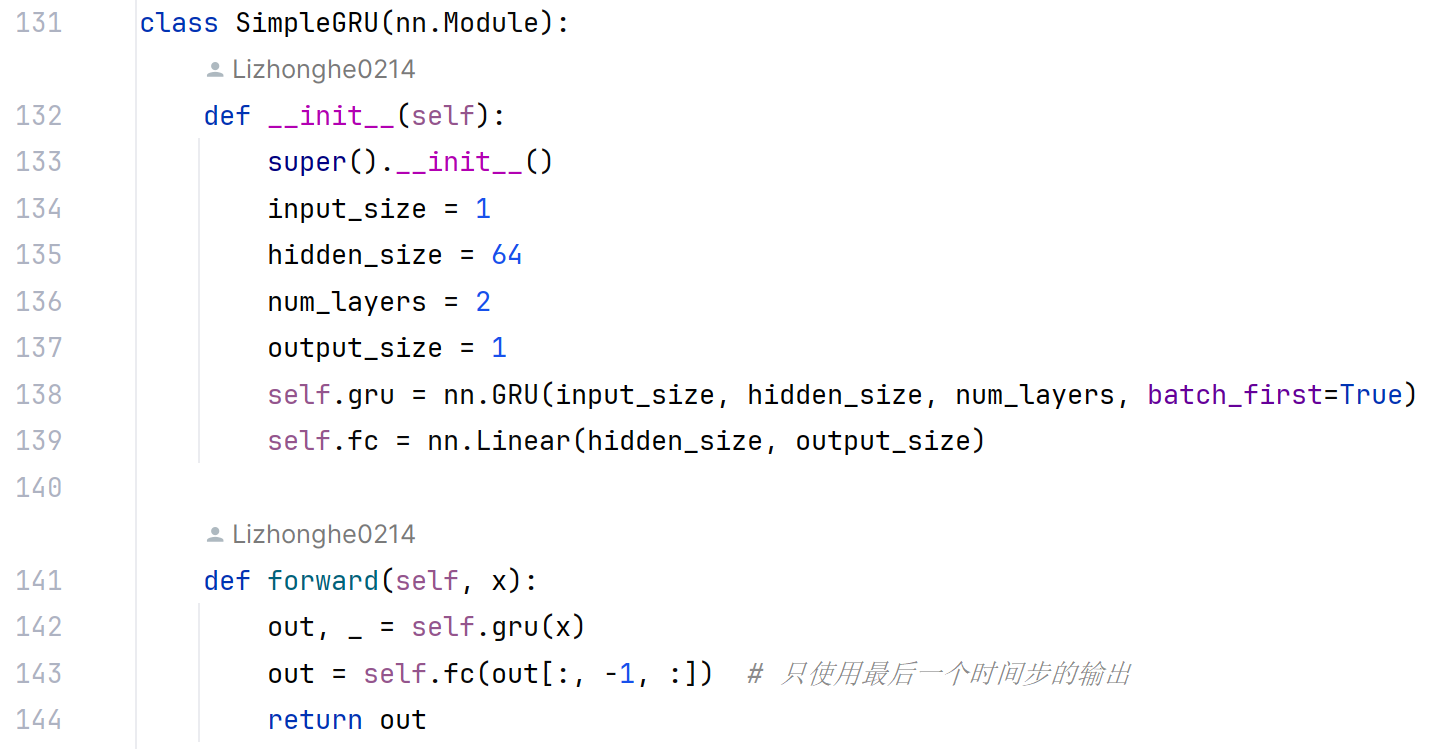


图 26 GRU模型定义代码

之后，实例化三种模型，并为每一个模型创建一个Adam优化器，其结合了AdaGrad和RMSProp的优点，能够自适应地调整每个参数的学习率，Adam优化器设置初始的学习率均为0.01。最后实例化均方误差损失函数和设置训练轮数为500次。

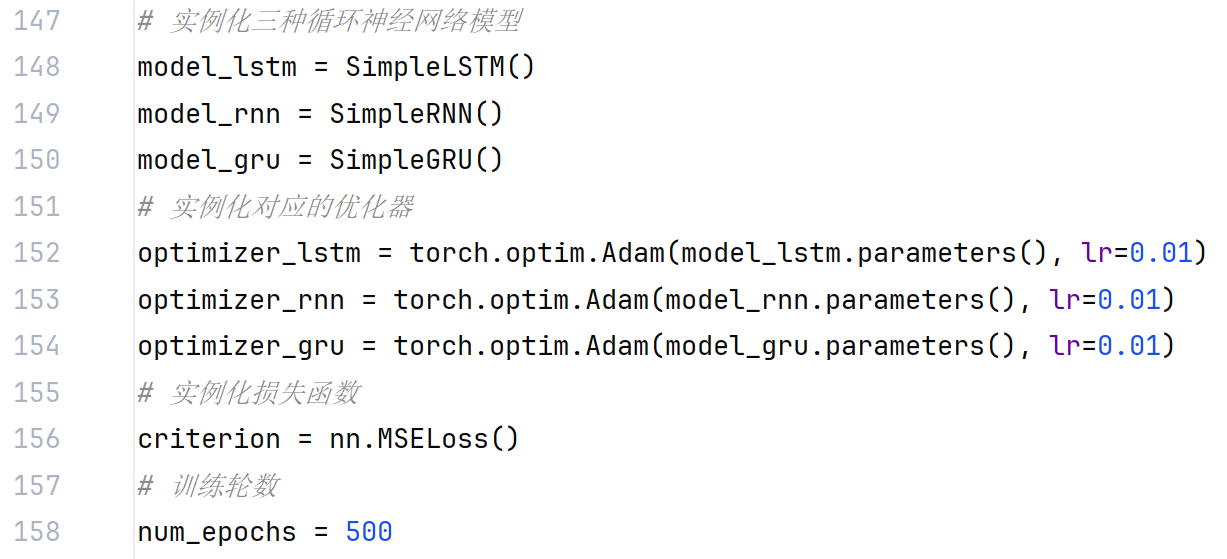


图 27模型实例化代码

3.6 训练模型

首先定义模型训练函数，训练过程中，每训练50个epoch，打印当前的训练进度和损失值，方便用户监控训练过程。接着定义模型预测函数。

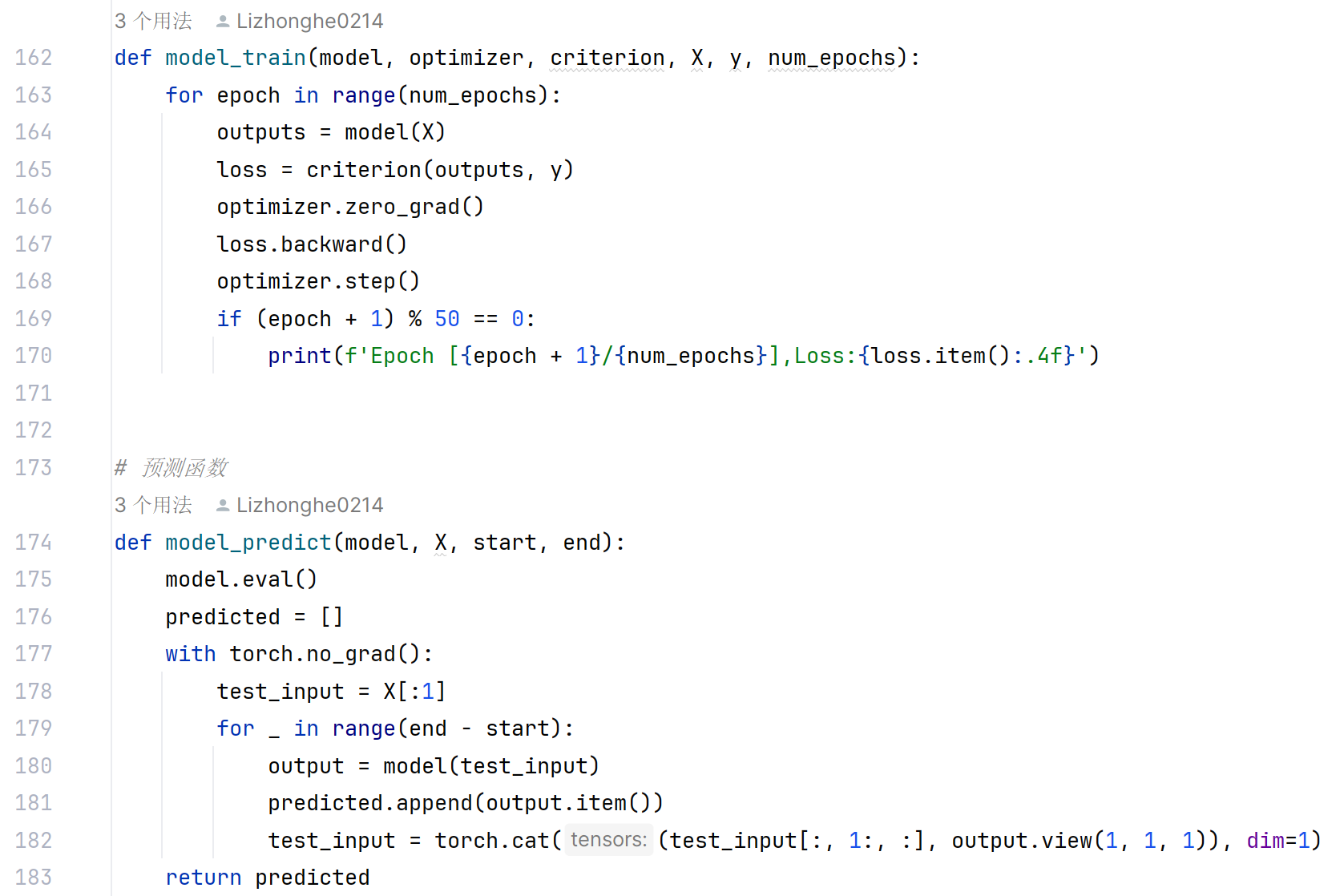


图 28模型训练和测试函数定义代码

3.7 模型预测结果可视化

首先训练LSTM模型，计算其预测值的均方误差MSE并可视化纽约市绿牌出租车费用的预测情况。

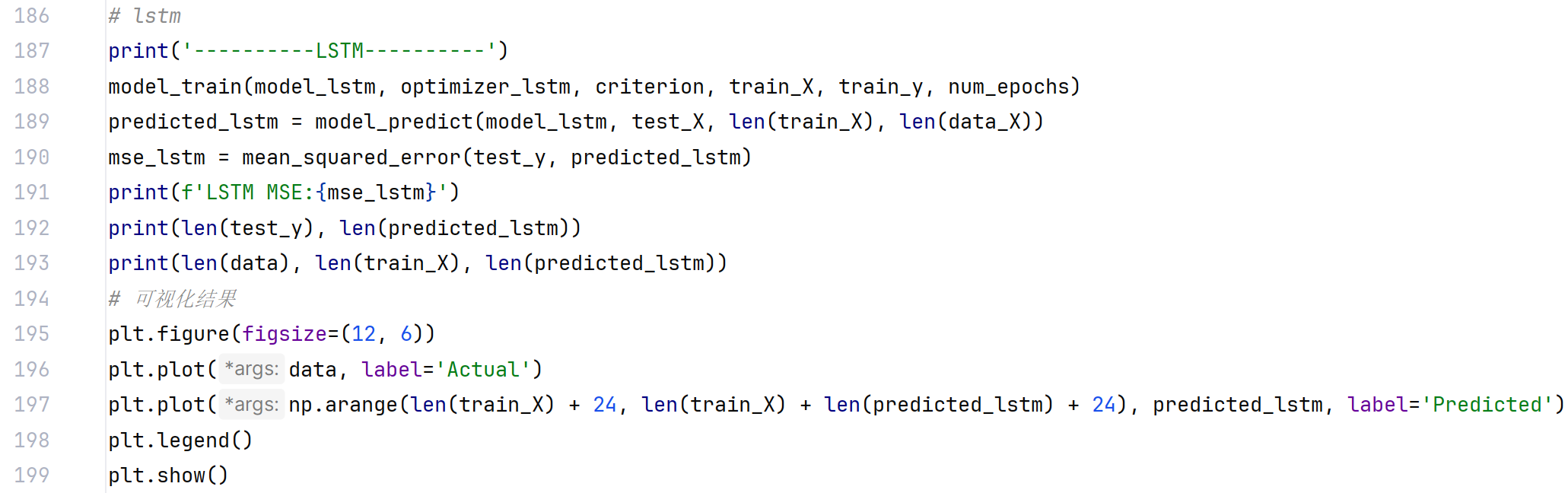


图 29 LSTM模型训练及预测代码

经计算，LSTM模型的均方误差MSE为0.01176。可视化的预测情况见图30，预测情况与实际情况基本吻合，除了个别峰值预测稍有偏差。这表明LSTM模型能够很好地学习训练数据，寻找出出租车费用流量的周期性变化规律。

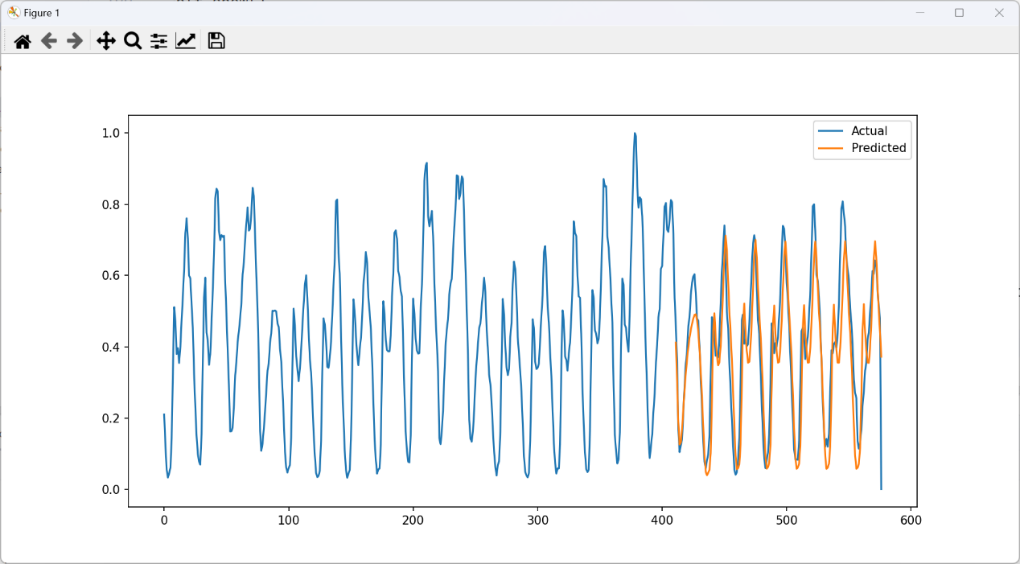


图 30 LSTM模型预测结果可视化

RNN模型的预测均方误差较大，为0.06622，可视化的预测情况见图31，周期性的规律能够拟合出来，但细节的峰值和次峰值没能很好地预测。

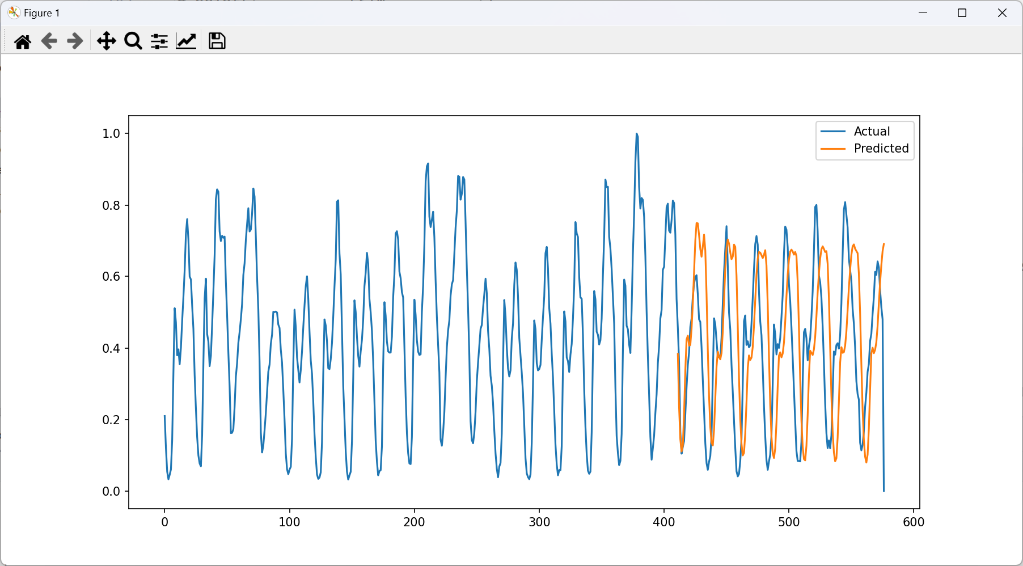


图 1 RNN模型预测结果可视化

GRU模型的预测均方误差为0.06448，可视化的预测情况见图32，周期性的规律及峰值与次峰值能够较好地预测，但预测的费用流量相比真实值整体稍有滞后，导致MSE偏大。



图 32 GRU模型预测结果可视化

1. 实验结论

经过对于scikit-learn的细致分析和上述实验的实践探索，归纳出一下sklearn库的显著优势：

1. 统一的API设计：sklearn库为各种机器学习算法提供了一致的接口，使得开发者可以快速上手不同的模型。无论使用分类、回归、聚类还是降维算法，基本的操作流程（如fit用于训练模型、predict用于预测）都是相似的，降低了学习成本和使用难度
2. 详细的文档和示例：sklearn库拥有丰富且详细的官方文档，每个类和函数都有清晰的说明和使用示例。文档中不仅解释了算法的原理和参数的含义，还提供了实际的代码示例，方便开发者参考和学习。同时，官方还提供了大量的教程和案例，帮助用户更好地理解和应用 sklearn 进行机器学习任务。
3. 涵盖多种机器学习任务：sklearn库提供了广泛的机器学习算法，涵盖了分类、回归、聚类、降维等多个领域，丰富的算法选择使得sklearn可以应用于各种不同类型的机器学习问题。
4. 算法实现质量高：sklearn库中的算法实现经过了严格的测试和优化，具有较高的性能和稳定性。开发者可以放心使用这些算法，而无需担心算法的正确性和效率问题。同时，sklearn还不断更新和改进算法，以适应新的研究成果和应用需求。
5. 支持多种数据格式：sklearn库可以处理多种常见的数据格式，如NumPy数组、Pandas数据框等。这使得开发者可以方便地使用不同来源的数据进行机器学习任务，无需进行复杂的数据格式转换。
6. 活跃的开源社区：sklearn库是一个开源项目，拥有庞大且活跃的社区。开发者可以在社区中分享经验、交流问题、贡献代码。社区中还会定期举办各种活动和讨论，促进sklearn的发展和应用。同时，社区也提供了丰富的学习资源和教程，帮助新手快速入门。
7. 与其他库的集成性好：sklearn库可以与其他流行的Python库（如NumPy、SciPy、Pandas、Matplotlib等）很好地集成。这种集成性使得开发者可以在一个统一的环境中完成整个机器学习流程，提高开发效率。

上述关于纽约市绿牌出租车费用流量预测的实践的完整代码和原始数据集可见于<https://github.com/Lizhonghe0214/Scikit-learn_Analysis>。此外，更多关于sklearn库的实操及使用技巧均可见与上述网址。