Python 大作业实验报告

WA2214014 杨跃浙

选题：基于 Python 的股票预测模型（Informer 模型应用于股票预测）

整体设计目标：

基于 Python 的股票预测模型的目标主要有：

1. 数据收集：从各种来源（如股票交易所、新闻网站、社交媒体等）收集与股票价格相关的数据，如历史价格、交易量、市盈率、市净率等。

2. 数据预处理：对收集到的数据进行清洗、缺失值处理、异常值处理等，以便后续分析和建模。

3. 特征工程：从原始数据中提取有用的特征，如技术指标（如移动平均线、相对强弱指数等）、基本面指标（如市盈率、市净率等）、市场情绪指标等。同时，可能需要进行特征选择，以减少噪声和提高模型性能。

4. 模型选择：根据问题的性质和数据特点，选择合适的预测模型，如线性回归、支持向量机、神经网络、随机森林等。

5. 模型训练：使用历史数据训练选定的模型，调整模型参数以优化预测性能。

6. 模型评估：使用测试数据集评估模型的预测性能，如计算预测误差、准确率、召回率等指标。

7. 模型优化：根据评估结果，对模型进行优化，如调整模型参数、增加或减少特征、尝试不同的模型等。

8. 实时预测：将优化后的模型应用于实时数据，进行股票价格预测。

9. 可视化：将预测结果以图表或其他形式展示，便于用户理解和分析。

10. 部署：将模型部署到生产环境，为用户提供股票预测服务。

本次实验主要以论文复现为主，由于时间有限本次实验主要做了其中“数据收集”，“模型选择”，“模型训练”，“模型评估”，“模型优化”，“可视化”部分。下面将结合 Python 外部库，介绍各个部分的主要功能和作用。
由于本项目较复杂，大量使用了外部库，所以之后将“代码总体框架”，“第三方库介绍”，“关键代码说明”三者结合起来按照实验设计目标来分块介绍。

数据收集：

本次数据收集主要依赖 tushare 库

Tushare 库介绍：Tushare 是一个免费、开源的 Python 财经数据接口包，主要实现对股票等金融数据从数据采集、清洗加工到数据存储的过程。该库专注于获取中国股票的历史和实时报价数据，易于使用，返回的数据大多为 pandas 的 DataFrame 对象，可以方便地保存为 csv, excel 或 json 文件，也可以插入到 MySQL 或 Mongodb 中。

Tushare 的数据品类丰富，包含股票、基金、期货、债券、外汇、行业大数据，以及数字货币行情等区块链数据。这些多样化的金融大数据可以为金融分析人员提供快速、整洁和多样的分析依据，极大地简化了数据获取过程。
源代码位于项目 getdata 文件夹中 getdata.py

代码主体框架：

def get_data()

函数get_data:把‘trade_date’(交易日)作为索引,导入七个维度‘close’,‘high’,‘low’,‘change’,‘vol’,‘amount’,‘open’,分别代表股票的收盘点位，最高点位，最低点位，涨跌点，成交量（手），成交额（千元），开盘点位，本次股票预测实验主要通过七个维度数据预测开盘价，即为七输入单输出模型。

def acquire_code()

函数 acquire_code：输入参数并调用 get_data 函数，实现数据获取，并将数据存储在 data 文件夹中 train.csv 中，需要注意的是为了之后训练，要把行索引名称改为‘date’，同时将股票数据按照时间正序排列。

```
df.sort_index(inplace=True)
df.index.name='date'
```

输入格式：

股票代码

example：600893.SH

数据起始、中止时间（格式：YYYYMMDD）

example：19981010(即 1998 年 10 月 10 日)

输出：文件格式：./data/train.csv

注意：ts.set_token()中参数为 tushare 库密钥，如果想要复现实验，需要去 tushare 官网注册并使用自己的 token 参数（不需要 pro 版本）官网有关于如何使用的详细介绍：https://tushare.pro/

模型选择：

本次实验训练模型主要依赖于 Informer 模型（AAAI 2021 Best Paper），Informer 模型在 2021 年提出后已获得的卓越的成果，并成为了 2021 年 AAAI 会议中 Best Paper 之一，论文已经下载在项目文件夹中，目前 Informer 模型已经在 GitHub 上开源并有详细的介绍：https://github.com/zhouhaoyi/Informer2020

Informer 模型论文：./Informer Beyond Efficient Transformer for Long Sequence Time-Series Forecasting.pdf

Informer 模型主要依赖 Pytorch 库实现，并送入 CUDA 加速。

Pytorch 库介绍：Pytorch 库极其强大，在这做简单介绍：PyTorch 是一个基于 Python 的开源机器学习库，由 Facebook's 人工智能研究团队于 2016 年发布，主要应用于构建和训练各种深度学习模型。

PyTorch 的核心组件是张量，这是一个用于存储和处理数据的多维数组。张量在 PyTorch 中起到了类似于 numpy 中的 ndarray 的角色，但是 PyTorch 的张量不仅可以在 CPU 上运行，还可以在 GPU 上运行，这为处理大量数据提供了可能。

除了其高效的计算能力，PyTorch 还以其灵活性著称。它允许使用 Python 这种在深度学习领域广泛使用的语言来表示深度学习模型，使得模型的构建和调试过程更为直观和便捷。无论是在学术研究还是在工业应用中，PyTorch 都得到了广泛的使用。

目前 Pytorch 已经更新至 2.1 版本，同时能很好的向下兼容，2.0 版本也增加了许多十分有趣的功能，并能更加适配 CUDA 加速，提高模型训练速度。

Pytorch 官网：https://pytorch.org/

（现在已经有 2.1 了我这边选择了 2.0 版本，如果要复现实验，一定要注意从官网装，conda 装出一大堆问题，后来被我删了用官网提供的 pip 指令成功安装）

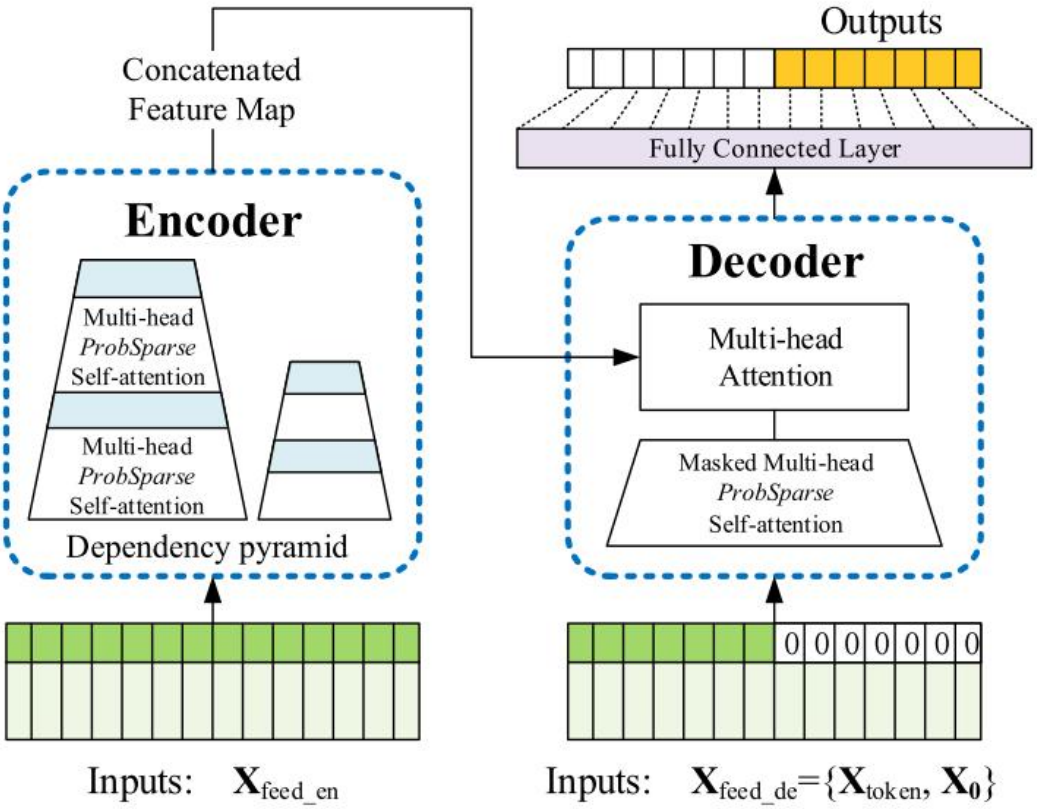
Argparse 库介绍：argparse 是 Python 官方提供的命令行参数解析库，其主要功能是解析命令行参数。

Informer 模型介绍：Informer 是一种基于 Transformer 的长序列预测模型，通过优化计算复杂度、内存使用和操作能力，实现了高效的时间复杂度和内存使用，以及高精度的预测能力。它采用了 ProbSparse Self-attention 机制、自注意力蒸馏机制和生成式解码器等改进方法，效果远超 Transformer。

在论文《Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting》中，作者详细介绍了 Informer 模型的背景、特点、结构和实验结果，并与其他相关工作进行了对比。此外，根据一些评论，这篇 AAAI2021 的论文获得了广泛的赞誉,其代码框架和论点的清晰度也得到了特别的认可。

Informer 模型不仅提高了预测精度和效率，还展示了强大的应用潜力，为长序列预测问题提供了新的解决方案。同时 Informer 保存了 Encoder-Decoder 的架构。

Informer 模型主体框架：



之后主要介绍用于训练该模型所需要修改的参数：

源代码位置：./main_informer.py

```
# 选择模型，我使用的 Informer 原始模型，Informerstack 是在 Informer 的基础上增加了一个 stack encoder，可以处理更长的输入序列，Informerlight(TBD)为轻量化 Informer
parser.add_argument('--model', type=str, default='informer',help='model of experiment, options: [informer, informerstack, informerlight(TBD)]')
# 选择数据类型，这里有 ETTh1，ETTh2，ETTm1，ETTm2，WTH 和 custom 的选择，其中前几个为官方给的数据类型，如果使用自己的数据集训练，则可选择 custom，但需要在本文件下面定义 custom
parser.add_argument('--data', type=str, default='custom', help='data')
# 选择进行时序预测的类型，M 为使用多变量时间序列作为输入，预测多变量的时间序列作为输出；S 为使用单变量时间序列作为输入，预测单变量的时间序列作为输出；MS 为使用多变量的时间序列作为输入，预测单变量的时间序列作为输出；
parser.add_argument('--features', type=str, default='MS', help='forecasting task, options:[M, S, MS]; M:multivariate predict multivariate, S:univariate predict univariate, MS:multivariate predict univariate')
# 这里的 target 参数即为在你的 csv 数据中你想预测的变量，比如我想预测开盘价，我
```

```
的 csv 文件中为 open，这里则是 open
parser.add_argument('--target', type=str, default='open', help='target feature in S or MS task')
# 设置时间特征编码，可以根据你训练集中的时间间隔设置，我这里设置的是天
parser.add_argument('--freq', type=str, default='d', help='freq for time features encoding, options:[s:secondly, t:minutely, h:hourly, d:daily, b:business days, w:weekly, m:monthly], you can also use more detailed freq like 15min or 3h')
# 权重保存位置
parser.add_argument('--checkpoints', type=str, default='./checkpoints/', help='location of model checkpoints')
#输入预测后的结果把下面的 store_true 改为 store_false，即可在./results/中的一个文件夹中找的 real_prediction.npy 文件，这里面存储的就是对未来的预测结果。
parser.add_argument('--do_predict', action='store_false', help='whether to predict unseen future data')
#Informer 有一个 early stop 机制，当训练过程中的 loss 值不在下降时，则会主动停止。
parser.add_argument('--patience', type=int, default=3, help='early stopping patience')
#股票预测模型较复杂，考虑增加训练轮数，防止欠收敛
parser.add_argument('--train_epochs', type=int, default=20, help='train epochs')
#修改训练以及测试集大小
parser.add_argument('--seq_len', type=int, default=400, help='input sequence length of Informer encoder')
parser.add_argument('--label_len', type=int, default=200, help='start token length of Informer decoder')
parser.add_argument('--pred_len', type=int, default=100, help='prediction sequence length')
```

模型训练：

该部分主要见视频

模型评估：

对于股票预测模型的评估主要可以根据 loss 函数或者训练集和测试集的拟合曲线差异，拟合曲线主要见可视化部分

模型优化：

考虑可以通过增加指标来增加时序预测的评判标准，当然最好的优化肯定是对数据进行充分处理，可以通过清洗，降噪，归一化等等操作，大大提升模

型训练准确度。但是本次实验主要考虑采用 Informer 模型去进行股票预测，故先不考虑数据的处理。

源代码位置：./utils/metrics.py

```
#各种时序预测指标
def MAE(pred, true):
    return np.mean(np.abs(pred-true))

def MSE(pred, true):
    return np.mean((pred-true)**2)

def RMSE(pred, true):
    return np.sqrt(MSE(pred, true))

def MAPE(pred, true):
    return np.mean(np.abs((pred - true) / true))

def MSPE(pred, true):
    return np.mean(np.square((pred - true) / true))

def RSE(pred, true):
    return np.sqrt(np.sum((true-pred)**2)) / np.sqrt(np.sum((true-true.mean())**2))
```

同时需要修改./exp/exp_informer.py 文件 221 行

```
mae, mse, rmse, mape, mspe,rse = metric(preds, trues)
print('mse:{}, mae:{},rmse:{},mape:{},mspe:{},rse:{}'.format(mse, mae,rmse,mape,mspe,rse))

np.save(folder_path+'metrics.npy', np.array([mae, mse, rmse,mape,rse]))
```

可视化（效果和结论）：

可视化主要依赖外部库 matplotlib,numpy,pillow 实现

Matplotlib 库介绍：Matplotlib 是一个 Python 2D 绘图库，它能够生成多种硬拷贝格式和跨平台的交互式环境，用于创建出版质量级别的图形。这个库被广泛应用于 Python 脚本、Python 和 IPython Shell、Jupyter 笔记本、Web 应用程序服务器以及四个图形用户界面工具包中。

Matplotlib 具有命令式、底层、可定制性强、图表资源丰富等特点，使得用户可以方便地生成各种类型的图像，如线图、散点图、等高线图、条形图、柱状图、3D 图形以及图形动画等。只需几行代码，就可以利用 Matplotlib 生成直方图、功率谱、错误图等多样化的图表。此外，Matplotlib 还提供了许多中级和高级教程以供参考和学习。

Matplotlib 不仅是一款功能强大的 Python 画图工具，也是一个综合库，用于在 Python 中创建静态、动画和交互式可视化。其丰富的功能和易用性使其成为 Python 绘图的首选库之一。

Numpy 库介绍：NumPy 库提供多维数组对象、各种派生对象以及用于数组快速操作的各种 API，包括数学、逻辑、形状操作、排序、选择、输入输出、离散傅立叶变换、基本线性代数，基本统计运算和随机模拟等等。此外，还提供了广播功能，支持对数组进行向量化运算。NumPy 是 Python 中常用的数值计算库，不仅支持多种数据类型和函数，而且提供快速的数值积分、线性代数运算和广播功能。这些特性使得 NumPy 在科学计算、数据分析、机器学习等领域有着广泛的应用。

Pillow 库介绍：Pillow 库，也被称为 PIL 库，是 Python 平台上常用的图像处理标准库，也是 Python3 最常用的图像处理库。Pillow 库支持的图片格式非常广泛，并且可以完成一些简单的图像处理任务。Pillow 库的主要功能和特点包括：读取和保存各种格式的图片；提供简洁易用的 API 接口，可以让您轻松地完成许多图像处理任务；进行图像的档案操作（如图片的批处理、缩略图绘制）、图像显示（在 Tkinter 框架下使用）、图像处理（例如点运算、卷积核滤波、色彩空间转化、大小变换、旋转、仿射变换）等。

源代码位置：./output/output.py

代码主体框架：

1. 数据导入
2. Test 集图像绘制
3. 函数 watermark_Image

def watermark_Image():

功能为添加个性化内容（水印）

需要注意的是要通过 Pillow 库设置字体大小必须采用自定义字体，默认字体无法改变字体大小，Pillow 支持加载 TrueType 和 OpenType 字体

TrueType 字体已经在 GitHub 上开源：

https://github.com/larsenwork/Gidole

我已经下载下来并放在项目文件

中：./Gidole-master/GidoleFont/Gidole-Regular.ttf

函数在图片左上角添加了“AHUniversity”“WA2214014 YangYuezhe”字样

```
drawing = ImageDraw.Draw(img)
black = (0, 0, 0)
textsize=25
ft=ImageFont.truetype(font='E:/Python/Project/Informer for Stock Prediction/Gidole-master/GidoleFont/Gidole-Regular.ttf',size=textsize)
drawing.text(pos, text, fill=black,font=ft,align="center")
```

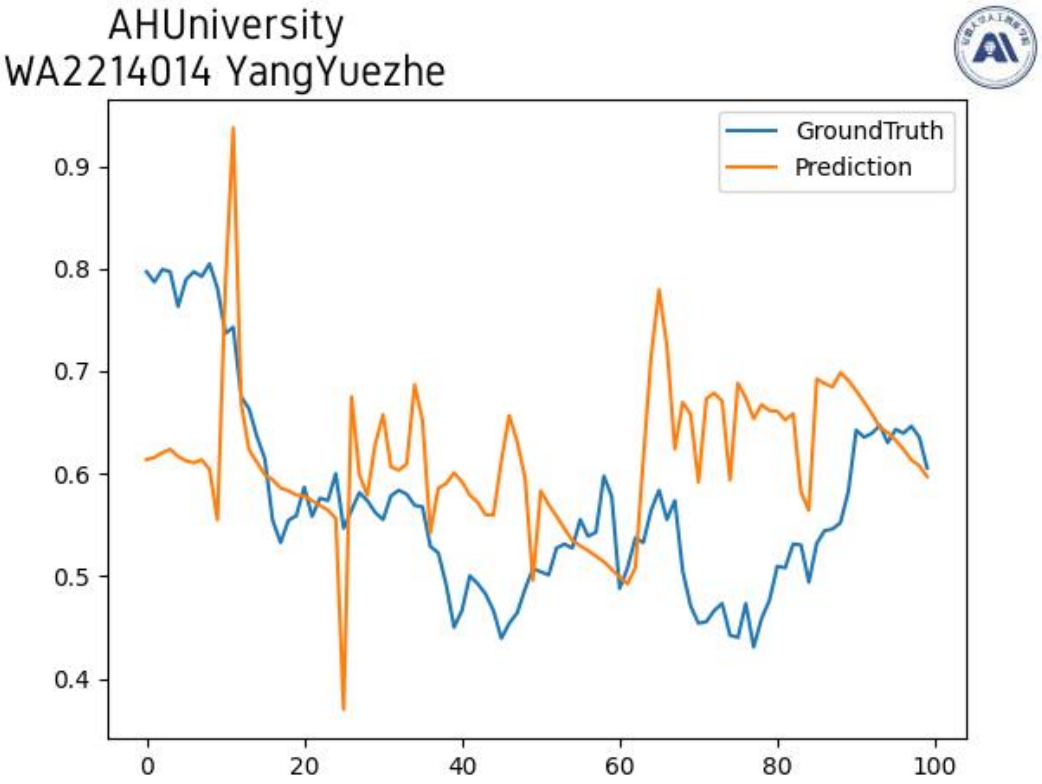
并在图片右上角添加了AI学院的图标 AI学院图标我已下载下来并放在项目文件中./AI.jpg

```
jpg=Image.open('E:/Python/Project/Informer for Stock Prediction/AI.jpg')
jpg_x=img.width-jpg.width-20
jpg_y=0
img.paste(jpg,(jpg_x,jpg_y))
```

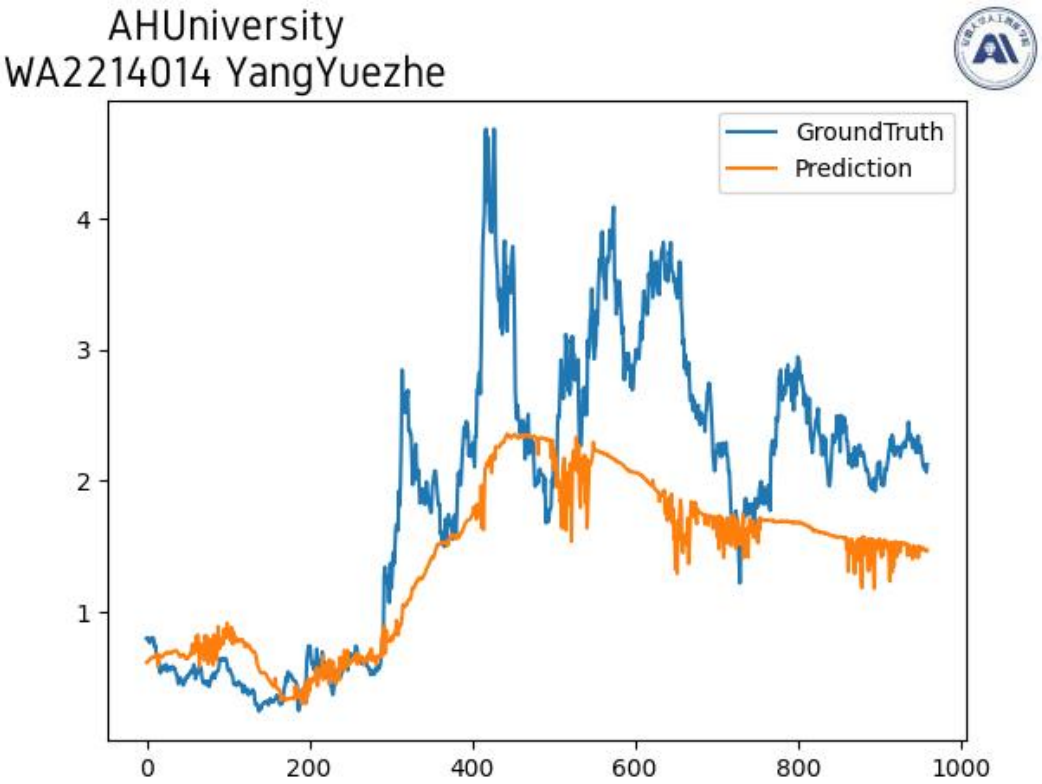
输入图像位置为./Stock Prediction.png (注意是 Informer for Stock Prediction 文件夹中的，并不是 output 文件夹中的)

4,调用 watermark_Image 函数

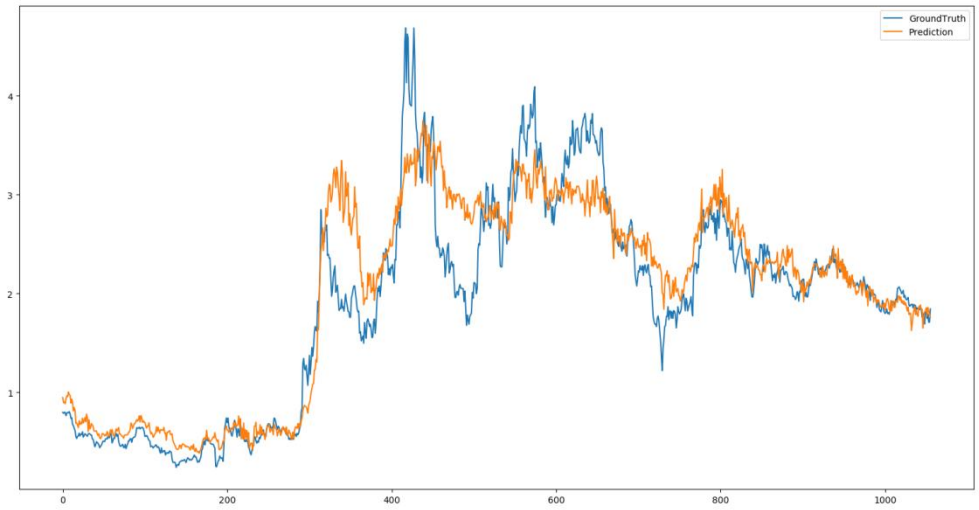
输出拟合的**测试集**结果：


AHUniversity
WA2214014 YangYuezhe
GroundTruth
Prediction
0.9
0.8
0.7
0.6
0.5
0.4
0
20
40
60
80
100

输出拟合的**训练集**结果：


AHUniversity
WA2214014 YangYuezhe
GroundTruth
Prediction
4
3
2
1
0
200
400
600
800
1000

这个给出一个效果比较好的训练集的拟合曲线（但是测试集的时候发现可能有**过拟合**的现象）



总结：

由于本次实验时间有限，本次股票模型预测主要依赖 Informer 模型实现，实际上已经有学者采用 Informer 和数据融合增强预处理，结合时事热点采用回归模型在股票预测上有了很不错的结果。本次实验不考虑数据处理方向，主要通过对模型参数等修改增加 Informer 模型对股票预测的准确度。本次实验主要创新点为引入 Informer 模型，与传统的 LSTM 模型对比，同时与 Transformer 模型对比，Informer 模型引入了多尺度时间注意力、概率自注意力机制、掩码机制、数据嵌入和位置编码等优化策略，使 Informer 能够更好地捕捉时间序列数据中的时序特征和上下文信息，Informer 模型在时序问题上有极好的表现，如今 Informer 模型已经被广泛应用于时序问题放预测上，如区域供热系统的负荷预测，基于信息员的电机轴承振动的时间序列预测，区域CO_2排放量的预测等等。但是在股票

预测上的应用仍较少，本次实验将 Informer 模型应用于股票预测，发现也有较好的表现，后续考虑可能通过加强数据处理来达到更可观的效果。

注：实验复现需要修改文件路径和申请密钥（token）！！

文中的./路径均表示文件夹中的 Informer for Stock Prediction 文件夹