

復旦大學

本科毕业论文（设计）



论文题目：基于机器学习的比特币价格预测及量化策略研究

姓名：曹肇立 学号：20307110028

院系：数学科学学院

专业：数学与应用数学

指导教师：李金凤 职称：讲师

单位：复旦大学数学科学学院

完成日期：2025 年 05 月 18 日

论文撰写人承诺书

本毕业论文是本人在导师指导下独立完成的，内容真实、可靠。本人在撰写毕业论文过程中不存在请人代写、抄袭或者剽窃他人作品、伪造或者篡改数据以及其他学位论文作假行为。

本人在撰写毕业论文过程中，如使用 AI 工具，将严格遵守学校及所属院系制定的使用规则，确保符合学术规范。（关于使用 AI 工具的信息披露：我在以下环境使用了人工智能工具辅助：(1)代码实现过程中借助 Kimi 辅助调试和优化模型结构；(2)论文写作阶段利用 Kimi 辅助格式规范化；(3)查重修改阶段使用 Kimi 辅助降低重复率。所有 AI 内容均经本人审核和修改，代码以及论文设计均由本人独立完成。使用的 Kimi 版本号为 v2.0.9）

本人清楚知道学位论文作假行为将会导致行为人受到不授予/撤销学位、开除学籍等处理（处分）决定。本人如果被查证在撰写本毕业论文过程中存在学位论文作假行为，愿意接受学校依法作出的处理（处分）决定。

承诺人签名：

苏泽云

日期：2025 年 05 月 18 日

指导教师对论文学术规范的审查意见：

本人经过尽职审查，未发现毕业论文有学术不端行为。

本人经过尽职审查，发现毕业论文有如下学术不端行为：

指导教师签名：

日期： 20 年 月 日

指导教师评语：

答辩委员会（小组）评语：

签名：
20 年 月 日

签名：
20 年 月 日

学分

成绩

备注：

教务处制

目 录

摘要	ii
Abstract	iii
第一章 绪论	2
1.1 研究背景与意义	2
1.1.1 比特币与加密货币市场概述	2
1.1.2 比特币价格波动特点及预测挑战	2
1.1.3 机器学习在价格预测中的应用前景	2
1.1.4 量化交易策略对投资决策的价值	3
1.2 文献综述	3
1.2.1 比特币定价机制与市场行为研究	3
1.2.2 比特币价格预测方法研究	3
1.2.3 比特币量化交易策略与市场效应研究	4
1.2.4 文献评述	4
1.3 研究内容与目标	4
1.4 研究思路与技术路线	5
1.5 论文结构安排	6
1.6 本研究的创新点	6
第二章 相关理论与技术基础	8
2.1 传统时间序列模型	8
2.1.1 ARIMA 模型	8
三元结构分解	8
模型构建流程	8
2.2 机器学习预测模型概述	9
2.2.1 梯度增强树模型(GBDT)	9
算法形式化描述	9
关键超参数与正则化	10
2.2.2 孤立森林(iForest)	10
算法原理	10
异常分数计算	11
2.2.3 K-近邻算法(KNN)	11
距离度量与邻域构建	11
预测函数与权重分配	12
2.2.4 极限梯度提升(XGBoost)	12

加法模型与目标函数	12
二阶近似优化	12
2.3 深度学习模型	13
2.3.1 长短期记忆网络(LSTM)	13
结构创新与信息流管理	13
2.4 量化交易基础	14
2.4.1 量化交易策略	14
2.4.2 策略回测与性能评估	14
第三章 数据选取与预处理	16
3.1 数据来源与描述	16
3.2 数据预处理	16
3.2.1 数据清洗	17
3.2.2 特征工程	17
3.2.3 数据归一化/标准化	17
3.2.4 数据集划分	18
第四章 基于机器学习的比特币价格预测模型构建与实证分析	19
4.1 实验环境配置	19
4.2 模型性能评价指标	19
4.3 比特币价格的动态复权机制	20
4.4 各预测模型构建与结果分析	22
4.4.1 ARIMA 模型预测结果	23
4.4.2 GBDT 模型预测结果	24
4.4.3 基于动态复权与 iForest-随机森林回归的优化预测模型	25
4.4.4 KNN 模型预测结果	27
4.4.5 LSTM 模型预测结果	28
4.4.6 加权优化 LSTM 模型预测结果	30
4.4.7 XGBoost 模型预测结果	31
4.5 模型性能综合对比与分析	32
4.5.1 模型优势与局限性分析	33
4.5.2 结论与启示	33
第五章 比特币量化交易策略设计与回测	35
5.1 量化交易策略设计思想	35
5.2 具体交易规则与参数设置	35
5.2.1 策略运作流程	35
5.2.2 关键参数设置	37
5.3 策略回测与绩效评估	38
5.3.1 回测设置	38
5.3.2 回测结果展示与分析	38

5.3.3 与经典量化策略的对比分析	39
经典策略的核心思路与交易逻辑	39
5.3.4 本研究策略的优越性分析	41
5.3.5 策略的优势与局限性	42
5.4 本章小结	43
第六章 总结与展望	44
6.1 研究工作总结	44
6.2 研究结论	44
6.3 研究不足与局限性	44
6.4 未来展望	45
附录 A 附录:量化交易策略 Python 代码	47
附录:量化交易策略 Python 代码	47

摘要

本研究聚焦于高波动性的比特币市场,旨在通过综合运用多种机器学习模型对比特币价格进行预测,并基于此设计与评估有效的量化交易策略。研究首先提出了一种创新性的比特币价格动态复权机制,有效解决了比特币供应量变动(尤其是“减半”事件)对价格序列连续性的影响。在此基础上,系统比较了 ARIMA、GBDT、KNN、LSTM 及 XGBoost 等模型在比特币价格预测任务中的性能,并开发了两种创新性优化方案:一是结合动态复权与 iForest 异常检测的随机森林回归预测模型,二是考虑比特币数量变动因素的 LSTM 加权优化方法。实验结果表明,基于动态复权的 iForest-随机森林模型在多个评价指标上展现出最优的预测能力,尤其在捕捉市场转折点方面表现突出。在此基础上,本研究设计了一套结合机器学习预测信号与技术指标的量化交易策略,并通过历史数据回测验证了其有效性。回测结果显示,该策略在特定参数设置下能够获得 2.704 的夏普比率、35.99% 的年化收益率,同时将最大回撤控制在 6.01% 的水平。本研究不仅为比特币价格预测提供了理论创新和模型优化思路,也为投资者在复杂数字货币市场中制定量化交易策略提供了具有实践价值的参考。

关键词:比特币;价格预测;机器学习;量化交易;时间序列分析;动态复权;LSTM;XGBoost

中图分类号:O24

Abstract

This research focuses on the highly volatile Bitcoin market, aiming to predict Bitcoin prices through comprehensive application of various machine learning models and to design and evaluate effective quantitative trading strategies based on these predictions. The study first proposes an innovative dynamic price adjustment mechanism for Bitcoin, effectively addressing the impact of Bitcoin supply changes (especially "halving" events) on price series continuity. On this foundation, the research systematically compares the performance of models such as ARIMA, GBDT, KNN, LSTM, and XGBoost in Bitcoin price prediction tasks, and develops two innovative optimization approaches: one combining dynamic price adjustment with iForest anomaly detection for random forest regression prediction, and another weighted optimization method for LSTM that considers Bitcoin quantity changes. Experimental results demonstrate that the iForest-random forest model based on dynamic price adjustment exhibits superior predictive capabilities across multiple evaluation metrics, particularly excelling in capturing market turning points. Building on this, the study designs a quantitative trading strategy that integrates machine learning predictive signals with technical indicators, and validates its effectiveness through historical data backtesting. The backtesting results show that under specific parameter settings, the strategy can achieve a Sharpe ratio of 2.704, an annualized return of 35.99%, while controlling the maximum drawdown to 6.01%. This research not only provides theoretical innovation and model optimization approaches for Bitcoin price prediction but also offers practical reference for investors formulating quantitative trading strategies in the complex digital currency market.

Keywords: Bitcoin; Price Prediction; Machine Learning; Quantitative Trading; Time Series Analysis; Dynamic Price Adjustment; LSTM; XGBoostt

CLC Code: O24

第一章 绪论

1.1 研究背景与意义

自 2009 年问世以来,比特币这种具有开创性的数字加密货币,其价格的剧烈波动和巨大的潜在回报吸引了全球投资者和研究者的广泛关注。相较于传统的金融资产,高波动性则在比特币市场中更为显著、更强的非线性和更复杂的驱动因素,这使得其价格预测成为一项极具挑战性的任务。准确预测比特币价格不仅对投资者制定投资策略、规避风险具有重要意义,也对理解新兴数字资产市场运行规律、维护金融市场稳定具有理论和现实价值。机器学习由于其所具有的非线性拟合能力以及能够从大数据中学习复杂模式的巨大优势,为比特币等加密货币价格预测提供了新的视角和有效的工具。通过构建基于机器学习的比特币价格预测模型,并结合有效的量化交易策略,有望在高度不确定的市场中捕捉盈利机会,提升投资组合的夏普比率,实现稳健的资产增值。因此,本研究旨在探索多种机器学习模型在比特币价格预测中的应用效果,并基于此设计和评估量化交易策略,为比特币市场的参与者提供决策参考。

1.1.1 比特币与加密货币市场概述

比特币(Bitcoin)是第一个使用点对点网络架构并以区块链技术为底层支撑的加密货币。它的诞生开启了数字货币的新纪元,并催生了庞大的加密货币市场。近年来,加密货币市场经历了快速发展和剧烈波动,其总市值曾一度达到数万亿美元。比特币作为市值最大、影响力最广的加密货币,其价格走势往往对整个加密货币市场具有引领作用。然而,比特币市场也存在诸多风险,如缺乏有效监管、易受市场情绪和突发事件影响、价格操纵风险等。深入理解比特币的特性及其所在市场的运行机制,是进行有效价格预测和策略研究的前提。

1.1.2 比特币价格波动特点及预测挑战

比特币价格呈现出典型的金融时间序列特征,包括高波动性、厚尾分布、波动聚集性以及潜在的非线性和非平稳性。影响比特币价格的因素众多且复杂,既包括宏观经济因素,也包括市场内部因素(如供需关系、投资者情绪、市场操纵行为、技术发展、监管政策变化)以及突发新闻事件等。这些因素相互交织,使得比特币价格预测面临巨大挑战。传统的线性时间序列模型(如 ARIMA)在处理这种高度复杂和非线性的数据时可能表现不佳,而机器学习模型则因其能够捕捉复杂依赖关系而受到青睐。

1.1.3 机器学习在价格预测中的应用前景

机器学习,尤其是深度学习,在预测问题上有着十分巨大的潜力。并且在金融领域,机器学习已被广泛应用于股票价格预测、信用风险评估、欺诈检测等多个方面。对于比特币价格预测,大量的机器学习模型,比如 iForest 抑或是 LSTM 等等,均能够从历史价格数据、交易量数据甚至文本数据(如新闻、社交媒体评论)中学习潜在规律,从而对未来价格走势进行预测。这些模型能够处理高维度数据,捕捉非线性关系,并具有一定的泛化能力,为提高比特币价格预测的准确性提供了可能。

1.1.4 量化交易策略对投资决策的价值

量化交易是运用数学、统计学和编程,从而判断出盈利的概率,并以此构建某种量化策略,以获取稳定超额收益的投资方式。在波动剧烈的比特币市场,基于可靠价格预测的量化交易策略能够帮助投资者克服情绪化交易的弱点,实现纪律性操作,并通过精确的入场点、出场点以及风险管理机制来提升投资绩效。设计并回测有效的量化交易策略,是检验价格预测模型实际应用价值的关键环节。

1.2 文献综述

本节旨在系统回顾与本研究主题相关的学术文献,主要涵盖比特币定价机制与市场行为、比特币价格预测方法(特别是机器学习与深度学习技术的应用)、以及基于这些预测的量化交易策略研究进展,作为本文的基准以及借鉴。

1.2.1 比特币定价机制与市场行为研究

加密货币市场的快速发展使其定价机制和市场行为成为研究热点。Shen, Urquhart, and Wang (2020) 针对超过 1700 种加密货币,做出了一种定价模型,其中包含了三个因子,即反转因子、规模因子和市场因子,发现该模型显著优于传统的加密货币资本资产定价模型(CCAPM),并指出交易量可能为加密货币市场提供信号,从而创造盈利策略^[1]。另一项关于《加密货币的资产定价》的研究通过时间序列分析,研究了诸如生产力或者是网络因子、甚至于投资者的关注度对加密货币收益的影响,结果表明网络因子和动量因子表现出显著的预测能力,而其构建的三因子模型(市场、规模、动量)能较好地解释加密货币市场的异常回报^[2]。此外,有研究采用贝叶斯结构时间序列方法,分析了比特币价格与黄金价格、人民币兑美元汇率、股票市场指数等多种内外因素的关系,揭示了比特币具有投机性、避险资产和潜在资本外逃工具的混合属性。

1.2.2 比特币价格预测方法研究

比特币价格预测是学术界和业界持续关注的焦点。传统的时间序列模型如 ARIMA 被广泛应用,但面对高度非线性的比特币价格数据时常显不足。ChChan and Majid (2018) 对比了人工神经网络(ANN)和 ARIMA 模型在比特币价格预测中的表现,发现在他们所研究的数据集和时间范围内,ANN 模型通常展现出更优的预测性能,尤其是在捕捉非线性动态方面^[3]。因此,机器学习与深度学习方法得到了越来越多的应用。研究者们尝试了包括 SVM、LSTM 在内的多种模型。

近年来,结合外部信息源(如社交媒体情绪)和先进深度学习架构的研究成为趋势。例如, Htay, Ghahremani, and Shaeles (2025) 提出通过整合社交媒体情绪(如 Twitter 文本数据的情绪分析结果和推文量)与历史价格数据来增强基于 LSTM 的比特币价格预测模型。他们的研究系统评估了这种多变量框架的益处,开发的 Multi-LSTM-Sentiment 模型在 MAE 和 RMSE 指标上均取得了最佳性能,强调了社交媒体情绪在预测建模中的重要性,并展示了其在高度动态的加密货币市场中提升决策能力的潜力^[4]。Saqr (2024) 借鉴机器人自适应行走技术,结合预训练的大型语言模型(LLM)和强化学习(RL),通过引入市场反馈信号来动态适应金融市场的变化,在应对市场状态变化方面表现出色^[5]。Xu et al. (2024) 则提出了一种使用深度强化学习(DRL)发现公式化阿尔法(交易信号)的新框架 ALPHA2,通过 DRL 引导的搜索算法在搜索空间中导航,优化评估指标,并确保阿尔法的逻辑一致性和多样性,实验证明该方法能够发现多样且有效的阿尔法,显著提高了最终交易策略的表现。

现^[6]。这些研究表明,先进的机器学习技术为提升比特币价格预测的精度和发现有效交易信号提供了新的途径。

1.2.3 比特币量化交易策略与市场效应研究

基于价格预测的量化交易策略是比特币研究的另一个重要方向。Padysak and Vojtko (2023) 对比特币市场的季节性效应、趋势跟随策略和均值回归策略进行了研究,发现比特币市场存在显著的季节性效应,尤其是在特定交易时段,且价格在达到局部极值后倾向于趋势延续或反转^[7]。Hanicová and Vojtko (2023) 研究了加密货币中的再平衡溢价现象,发现定期再平衡投资组合可以显著提高累计回报率并降低波动率,尤其是在结合低波动资产时效果更佳^[8]。Vojtko and Javorská (2024) 进一步深入研究了比特币的季节性效应,提出了一个简单的季节性交易策略,即每天仅在特定的两个小时持有比特币,并发现其回报率显著高于其他时段,尤其是在市场上升趋势和特定星期几(如周五)^[9]。然而,Mueller (2024) 重新审视了加密货币市场的季节性效应(如周一效应、周末效应等),发现这些效应并不稳健,且随时间推移而变化,提示投资者需警惕策略的失效风险^[10]。Huang, Sangiorgi, and Urquhart (2024) 则研究了基于交易量加权的时间序列动量(TSMOM)策略,发现该策略能够显著超越其他投资组合,并且在扣除交易成本后仍具有盈利能力,其收益无法完全通过已知的加密货币三因子模型解释^[11]。这些研究为设计和评估比特币量化交易策略提供了丰富的思路和实证依据。

1.2.4 文献评述

综上所述,国内外学者在比特币等加密货币的定价机制、价格预测方综上,国内外学者对比特币的定价、预测及量化交易策略进行了深入研究,发现其价格受多种因素影响,波动复杂。机器学习和深度学习方法因非线性拟合能力强,在比特币价格预测中优于传统模型,展现出更大应用潜力。法以及量化交易策略方面已经开展了广泛而深入的研究,并取得了一系列有价值的成果。现有研究表明,比特币价格受到多种复杂因素的影响,其波动呈现出典型的非线性、高波动性特征。机器学习深度学习方法因其强大的非线性拟合能力,在比特币价格预测领域展现出较传统计量模型更优的性能和更广阔的应用前景。同时,基于这些预测模型的量化交易策略研究也日益丰富,为投资者提供了多样化的决策支持。

然而,现有研究仍存在一些可拓展的空间。首先,多数研究侧重于单一预测模型的应用,对于多种模型(尤其是结合传统模型、集成学习模型和深度学习模型)的系统性比较和融合研究尚不充分。其次,在模型优化方面,如何更有效地结合比特币市场的特性(如本研究中考虑的比特币数量变动对LSTM的优化)进行针对性改进,仍有探索价值。再次,虽然已有不少关于量化策略的研究,但如何设计出在真实市场环境下(考虑交易成本、滑点等因素)依然稳健且具有较高风险调整后收益(如高夏普比率)的策略,仍是持续的挑战。最后,针对特定机器学习模型(如 iForest)在比特币价格预测流程中的具体应用场景(如作为异常数据处理、特征选择工具或直接预测模型)及其效果的探讨也有待深化。

本研究正是在上述背景下展开,旨在系统比较多种主流机器学习模型在比特币价格预测中的表现,探索基于比特币特有属性的模型优化方法,并最终设计和验证一个基于预测结果的量化交易策略,以期为比特币市场提供新观察。

1.3 研究内容与目标

本研究的主要内容包括以下几个方面:

- (1) 数据收集与预处理: 收集比特币的历史价格数据、交易量数据以及用于 LSTM 模型优化的比特币数量变动数据, 并进行必要的数据清洗、特征工程和数据集划分。
- (2) 机器学习预测模型构建与比较: 分别构建并训练 ARIMA、GBDT、iForest(需明确其具体应用, 例如作为异常检测或特征筛选)、KNN、LSTM 和 XGBoost 等多种机器学习模型, 用于预测比特币的未来价格走势。
- (3) LSTM 模型优化: 针对 LSTM 模型, 考虑比特币的实际流通数量或供应量变化因素, 进行加权优化, 以期提升预测精度。
- (4) 模型性能评估: 采用 MSE、MAE、R² 三大指标来为预测模型的性能进行综合评估和比较, 筛选出表现最优的预测模型。
- (5) 量化交易策略设计与回测: 基于最优的比特币价格预测模型, 设计一套完整的量化交易策略, 包括明确的买入、卖出信号, 以及止盈止损机制。利用历史数据对该策略进行回测, 评估其盈利能力和平风险水平, 特别是计算夏普比率等关键绩效指标。

本研究的预期目标是:

- (1) 成功构建并验证多种机器学习模型在比特币价格预测任务中的有效性, 并找出相对最优的预测模型。
- (2) 验证基于比特币数量变动对 LSTM 模型进行加权优化的有效性。
- (3) 设计一个具有实际操作价值的比特币量化交易策略, 并通过回测证明其能够获得理想的风险调整后收益。
- (4) 为比特币投资提供有益参考。

1.4 研究思路与技术路线

本研究的技术路线如下图所示:

- (1) 数据层: 首先, 从可靠的数据源获取比特币日度或更高频率的历史价格数据以及比特币流通量数据。首先进行数据清洗, 随后则是进行特征工程, 可能包括计算常用的技术指标, 并将时间序列数据转换为适用于监督学习的格式。
- (2) 模型层:
 - 1) 基准模型: 选择 ARIMA 作为传统时间序列预测模型的基准。
 - 2) 机器学习模型: 分别实现并训练 GBDT、KNN、XGBoost 等集成学习和 K-近邻算法模型。
 - 3) 深度学习模型: 构建 LSTM 网络模型, 利用其捕捉时间序列长期依赖的能力。特别地, 将根据比特币数量的变动信息对 LSTM 模型的输入或损失函数进行加权优化, 以反映比特币稀缺性动态变化对价格的影响。
- (3) 策略层: 基于在模型层筛选出的最优预测模型, 设计量化交易策略。策略将明确定义交易信号的产生条件、仓位管理规则、以及止盈和止损条件。
- (4) 评估层:
 - 1) 模型评估: 使用 MSE、MAE、RMSE、R² 等指标评估各预测模型的预测精度。对比分析不同模型以及优化后 LSTM 模型的性能。
 - 2) 策略评估: 对设计好的量化交易策略进行历史回测。计算关键绩效指标, 如最大回撤、年化收益率、胜率、以及夏普比率。
- (5) 结论与展望: 总结研究发现, 讨论模型的优缺点、策略的有效性及局限性, 并对未来研究方向进行

展望。

1.5 论文结构安排

本论文共分为五章,具体安排如下:

第一章:绪论。本章聚焦于研究背景与意义(涵盖比特币定价机制与市场行为、价格预测方法、量化交易策略以及文献评述)。此外,阐述研究内容与目标,明晰研究思路与技术路径,勾勒论文结构,并点明创新点。

第二章:相关理论与技术基础。详细介绍本研究中涉及的传统时间序列模型(ARIMA)、各类机器学习预测模型(GBDT、iForest、KNN、XGBoost)、深度学习模型(LSTM 及其优化思路)以及量化交易的基本概念和理论。

第三章:基于机器学习的比特币价格预测模型构建与实证分析。详细阐述实验数据的选取与预处理方法、实验环境配置、模型性能评价指标,并对 ARIMA、GBDT、iForest、KNN、LSTM、XGBoost 以及加权优化的 LSTM 等各个模型的构建过程、参数设置、训练结果进行详细分析,最后对各模型的预测性能进行综合对比。

第四章:比特币量化交易策略设计与回测。主要阐述基于最优预测模型的量化交易策略的设计思想、具体交易规则与参数设置、回测环境与过程,并对策略的回测结果以及种种指标进行深入分析与评估。

第五章:总结与展望。总结全文的研究工作和主要结论,在总结主要成果的同时,对研究中的不足与潜在局限性进行了客观剖析。此外,对未来研究方向及拓展应用的可能性进行了前瞻性的探讨。

1.6 本研究的创新点

本论文研究的创新点包括:

- (1) **比特币价格动态复权机制设计**:针对比特币供应量随时间指数衰减的特性(尤其是“减半”事件),本研究首创性地借鉴股票市场“动态复权”思想,设计了适用于比特币的价格复权机制。该机制有效消除了因供应量变动导致的价格“跳空”或“失真”,为预测模型提供了更能反映比特币内在价值连续变动的价格序列,是本研究的重要理论创新。
- (2) **iForest 与动态复权的创新结合**:本研究独创性地将孤立森林(iForest)异常检测算法与动态复权机制相结合,构建了一个“复权-异常检测-随机森林回归”的完整预测流程。该方法不仅能有效识别并处理复权价格空间中的异常点,还能在保留市场真实波动信息的同时过滤噪声,显著提升了预测精度,尤其在捕捉市场转折点方面表现出色。实验结果证明,这一创新组合是本研究中表现最优的预测方案。
- (3) **基于比特币特性的 LSTM 模型优化**:针对 LSTM 模型,本研究创新性地引入了比特币数量变动(如流通量或挖矿产出变化)作为加权因素,设计了输入特征加权和损失函数加权两种优化策略。这种从比特币供给层面优化模型的方法,显著提升了 LSTM 对比特币价格波动的捕捉能力,为深度学习模型在加密货币预测中的应用提供了新思路。
- (4) **高夏普比率量化策略的实证**:基于优选的预测模型,本研究设计并回测了一个结合机器学习预测信号与技术指标的量化交易策略。通过实证分析,验证了该策略能够达到 2.704 的夏普比率和 35.99% 的年化收益率,同时将最大回撤控制在 6.01% 的水平,为比特币市场的实际投资操作提

供了具有较强参考价值的策略方案。

- (5) **研究视角的整合性**: 本研究不仅关注单一的价格预测精度,更将预测模型与实际的量化交易策略设计紧密结合,形成从理论创新(动态复权机制)到模型优化(iForest 与 LSTM 改进)再到实践应用(量化交易策略)的完整研究链条,强调了预测研究的最终落脚点在于提升投资决策的有效性。

第二章 相关理论与技术基础

本章主要介绍本研究中所涉及到的核心理论与技术方法,包括传统的时间序列分析模型、各类机器学习预测算法、深度学习网络结构以及量化交易的基本概念,为后续的模型构建和策略设计奠定理论基础。

2.1 传统时间序列模型

2.1.1 ARIMA 模型

自回归积分滑动平均模型(ARIMA)通过三重数学机制的耦合实现对序列动态特性的刻画。

三元结构分解

ARIMA(p, d, q) 模型的数学表征可分解为三个互补组件:

(1) **自回归项 $\mathcal{AR}(p)$** : 构建当前观测值与其历史值间的线性依存关系, 形式化表达为:

$$\mathcal{AR}(p) : X_t = \alpha + \sum_{i=1}^p \varphi_i X_{t-i} + \varepsilon_t \quad (2.1)$$

其中 α 为截距项, φ_i 为自回归系数, $\varepsilon_t \sim \mathcal{N}(0, \sigma^2)$ 为高斯白噪声过程。

(2) **差分环节 $\mathcal{I}(d)$** : 通过迭代差分运算消除非平稳性, d 阶差分定义为:

$$\nabla^d X_t = (1 - B)^d X_t \quad (2.2)$$

其中 B 为滞后算子, 满足 $BX_t = X_{t-1}$, $(1 - B)^d$ 表示 d 次迭代应用一阶差分算子。

(3) **滑动平均项 $\mathcal{MA}(q)$** : 捕捉随机冲击的持续效应, 其数学形式为:

$$\mathcal{MA}(q) : X_t = \mu + \varepsilon_t + \sum_{j=1}^q \theta_j \varepsilon_{t-j} \quad (2.3)$$

其中 μ 为过程均值, θ_j 为滑动平均权重系数。

综合上述三元素, 完整的 ARIMA(p, d, q) 模型可表述为:

$$\phi(B)(1 - B)^d X_t = \theta(B)\varepsilon_t \quad (2.4)$$

其中 $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p$ 和 $\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q$ 分别为自回归和滑动平均多项式。

模型构建流程

ARIMA 建模遵循以下环节:

- i. **平稳性诊断:** 通过时序图谱分析、自相关函数(ACF)衰减特性考察及单位根检验(如增广Dickey-Fuller检验)确定差分阶数 d 。平稳化后的序列应满足:

$$\mathbb{E}[X_t] = \text{常数}, \quad \text{Var}[X_t] = \text{常数}, \quad \text{Cov}[X_t, X_{t+h}] = f(h) \quad (2.5)$$

- ii. **参数阶次识别:** 基于平稳序列的自相关函数(ACF)与偏自相关函数(PACF)的截尾/拖尾模式确定 p 和 q 值。典型判据为:

$$\text{PACF}(k) \approx 0 \text{ for } k > p \Rightarrow \text{AR}(p) \quad (2.6)$$

$$\text{ACF}(k) \approx 0 \text{ for } k > q \Rightarrow \text{MA}(q) \quad (2.7)$$

亦可借助信息准则(如 AIC、BIC)进行模型选择:

$$\text{AIC} = -2 \ln(L) + 2k, \quad \text{BIC} = -2 \ln(L) + k \ln(n) \quad (2.8)$$

其中 L 为似然函数值, k 为参数数量, n 为样本量。

- iii. **参数估计:** 采用最大似然估计(MLE)或条件最小二乘法(CLS)求解模型参数。对于高斯过程, 对数似然函数为:

$$\ln L(\phi, \theta, \sigma^2) = -\frac{n}{2} \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{t=1}^n \varepsilon_t^2 \quad (2.9)$$

- iv. **诊断验证:** 对残差序列 $\{\hat{\varepsilon}_t\}$ 进行白噪声检验, 如 Ljung-Box Q 统计量:

$$Q(m) = n(n+2) \sum_{k=1}^m \frac{\hat{\rho}_k^2}{n-k} \sim \chi^2_{m-p-q} \quad (2.10)$$

其中 $\hat{\rho}_k$ 为残差的样本自相关系数, m 为滞后阶数。

- v. **预测外推:** 基于估计模型生成 h 步前向预测:

$$\hat{X}_{n+h|n} = \mathbb{E}[X_{n+h} | \mathcal{F}_n] \quad (2.11)$$

其中 \mathcal{F}_n 表示截至时刻 n 的信息集。

2.2 机器学习预测模型概述

机器学习算法通过从数据中学习模式和规律, 从而对未知数据进行预测或分类。本研究采用了多种具有代表性的机器学习模型, 它们在处理非线性关系和复杂数据方面通常优于传统线性模型。

2.2.1 梯度增强树模型(GBDT)

梯度增强决策树(GBDT)通过序贯构建基学习器并优化残差逼近, 实现高精度非线性映射。其数学本质可表述为在函数空间中沿损失函数负梯度方向的迭代优化过程。

算法形式化描述

给定训练集 $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, GBDT 通过以下递推形式构建加性模型:

$$F_M(x) = \sum_{m=0}^M \gamma_m h_m(x) \quad (2.12)$$

其中 $F_0(x) = \arg \min_c \sum_{i=1}^n L(y_i, c)$ 为初始常数预测器, $h_m(x)$ 为第 m 轮基学习器 (通常为 CART 决策树), γ_m 为步长系数。

模型训练遵循前向分步算法, 第 m 轮迭代目标为:

$$(h_m, \gamma_m) = \arg \min_{h, \gamma} \sum_{i=1}^n L(y_i, F_{m-1}(x_i) + \gamma h(x_i)) \quad (2.13)$$

由于上述优化问题难以直接求解, GBDT 采用函数梯度近似策略, 即:

$$r_{im} = - \left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)} \right]_{F=F_{m-1}} \quad (2.14)$$

$$h_m = \arg \min_h \sum_{i=1}^n (r_{im} - h(x_i))^2 \quad (2.15)$$

$$\gamma_m = \arg \min_{\gamma} \sum_{i=1}^n L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)) \quad (2.16)$$

其中 r_{im} 为负梯度残差, h_m 通过拟合残差构建, γ_m 通过线搜索确定。

关键超参数与正则化

GBDT 性能受多维超参数调控, 主要包括:

- i. 学习率 $\eta \in (0, 1]$: 控制每轮贡献权重, 实际更新为 $F_m = F_{m-1} + \eta \gamma_m h_m$
- ii. 树深度 d_{max} : 限制单棵树复杂度, 平衡拟合能力与泛化性
- iii. 子采样率 $s \in (0, 1]$: 每轮随机抽取 $s \cdot n$ 样本构建树, 引入随机性增强鲁棒性
- iv. 正则化项 $\Omega(h)$: 树结构复杂度惩罚, 如叶节点数量或 L2 范数约束

综合优化目标可表示为:

$$\mathcal{L} = \sum_{i=1}^n L(y_i, F_M(x_i)) + \sum_{m=1}^M \Omega(h_m) \quad (2.17)$$

2.2.2 孤立森林(iForest)

孤立森林作为一种基于随机子空间分割的异常检测算法, 其核心思想源于异常样本在特征空间中的稀疏性特征, 即异常点通常更易被随机划分过程“孤立”。

算法原理

孤立森林通过构建多棵随机分割树来检测异常。每棵树的构建过程如下:

$$\text{构建树}(\mathcal{X}) = \begin{cases} \text{叶节点,} & \text{若 } |\mathcal{X}| \leq 1 \text{ 或达到最大深度} \\ \text{分割}(\mathcal{X}, a, \theta), & \text{否则} \end{cases} \quad (2.18)$$

其中分割操作为：

$$a \sim \text{随机选择特征} \quad (2.19)$$

$$\theta \sim \text{特征 } a \text{ 上的随机分割点} \quad (2.20)$$

$$\mathcal{X}_L = \{\mathbf{x} \in \mathcal{X} \mid \mathbf{x}_a < \theta\} \quad (2.21)$$

$$\mathcal{X}_R = \{\mathbf{x} \in \mathcal{X} \mid \mathbf{x}_a \geq \theta\} \quad (2.22)$$

异常分数计算

对于样本 \mathbf{x} , 其异常分数定义为:

$$s(\mathbf{x}) = 2^{-\frac{h(\mathbf{x})}{c(n)}} \quad (2.23)$$

其中:

$$h(\mathbf{x}) = \text{样本 } \mathbf{x} \text{ 在森林中的平均路径长度} \quad (2.24)$$

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n} \approx 2\ln(n-1) - \frac{2(n-1)}{n} \quad (2.25)$$

$H(i)$ 为调和数, n 为训练样本数。分数解释:

$$s(\mathbf{x}) \rightarrow 1 : \text{强异常} \quad (2.26)$$

$$s(\mathbf{x}) \approx 0.5 : \text{边界样本} \quad (2.27)$$

$$s(\mathbf{x}) \ll 0.5 : \text{正常样本} \quad (2.28)$$

2.2.3 K-近邻算法(KNN)

K-近邻 (K-NN) 算法作为一种基于实例的非参数学习方法, 通过局部邻域信息推断未知样本属性。

距离度量与邻域构建

给定训练集 $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n \subset \mathbb{R}^d \times \mathbb{R}$, K-NN 回归通过以下步骤实现预测:

首先, 定义距离函数 $\mathcal{D} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^+$, 常用的 p -范数距离为:

$$\mathcal{D}_p(x, z) = \|x - z\|_p = \left(\sum_{j=1}^d |x_j - z_j|^p \right)^{1/p} \quad (2.29)$$

特殊情况包括:

- (1) $p = 1$: 曼哈顿距离 $\mathcal{D}_1(x, z) = \sum_{j=1}^d |x_j - z_j|$
- (2) $p = 2$: 欧几里得距离 $\mathcal{D}_2(x, z) = \sqrt{\sum_{j=1}^d (x_j - z_j)^2}$
- (3) $p = \infty$: 切比雪夫距离 $\mathcal{D}_\infty(x, z) = \max_{j \in \{1, \dots, d\}} |x_j - z_j|$

对于测试样本 x_q , 其 K 近邻集合定义为:

$$\mathcal{N}_K(x_q) = \{(x_i, y_i) \in \mathcal{D} \mid |\{(x_j, y_j) \in \mathcal{D} \mid \mathcal{D}(x_q, x_j) < \mathcal{D}(x_q, x_i)\}| < K\} \quad (2.30)$$

预测函数与权重分配

基于构建的邻域, K-NN 回归的预测函数可表示为:

$$f(x_q) = \sum_{(x_i, y_i) \in \mathcal{N}_K(x_q)} w_i \cdot y_i \quad (2.31)$$

其中权重 w_i 可采用多种分配策略:

- i. 均匀权重: $w_i = \frac{1}{K}$
- ii. 距离倒数权重: $w_i = \frac{\mathcal{D}(x_q, x_i)^{-1}}{\sum_{(x_j, y_j) \in \mathcal{N}_K(x_q)} \mathcal{D}(x_q, x_j)^{-1}}$
- iii. 核权重: $w_i = \frac{K(x_q, x_i)}{\sum_{(x_j, y_j) \in \mathcal{N}_K(x_q)} K(x_q, x_j)}$, 其中 $K(\cdot, \cdot)$ 为核函数

2.2.4 极限梯度提升(XGBoost)

XGBoost (eXtreme Gradient Boosting) 作为梯度提升决策树的高效实现, 通过一系列算法优化与工程改进, 在结构化数据建模领域展现出卓越性能。本节从函数逼近与优化理论角度阐述其数学基础。

加法模型与目标函数

XGBoost 基于加法模型构建预测函数, 对于样本 \mathbf{x}_i , 其预测值在第 t 轮迭代后表示为:

$$\hat{y}_i^{(t)} = \sum_{k=1}^t f_k(\mathbf{x}_i) = \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i) \quad (2.32)$$

其中 $f_k \in \mathcal{F}$ 为基学习器, \mathcal{F} 为回归树函数空间。XGBoost 的核心创新在于其目标函数设计:

$$\mathcal{L}^{(t)} = \sum_{i=1}^n \ell(y_i, \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)) + \Omega(f_t) \quad (2.33)$$

其中 $\ell(\cdot, \cdot)$ 为可微损失函数, $\Omega(f_t)$ 为正则化项:

$$\Omega(f_t) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2 \quad (2.34)$$

T 表示叶节点数量, w_j 为第 j 个叶节点的权重, γ 和 λ 为正则化系数。

二阶近似优化

XGBoost 通过对目标函数进行二阶泰勒展开实现高效优化:

$$\mathcal{L}^{(t)} \approx \sum_{i=1}^n \left[\ell(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i) \right] + \Omega(f_t) \quad (2.35)$$

其中 g_i 和 h_i 分别为一阶和二阶梯度:

$$g_i = \frac{\partial \ell(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}_i^{(t-1)}} \quad (2.36)$$

$$h_i = \frac{\partial^2 \ell(y_i, \hat{y}_i^{(t-1)})}{\partial (\hat{y}_i^{(t-1)})^2} \quad (2.37)$$

忽略常数项,目标函数可简化为:

$$\tilde{\mathcal{L}}^{(t)} = \sum_{i=1}^n \left[g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i) \right] + \Omega(f_t) \quad (2.38)$$

定义树结构 $q : \mathbb{R}^d \rightarrow \{1, 2, \dots, T\}$,将样本映射到叶节点,则:

$$f_t(\mathbf{x}_i) = w_{q(\mathbf{x}_i)} \quad (2.39)$$

目标函数可重写为:

$$\tilde{\mathcal{L}}^{(t)} = \sum_{j=1}^T \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \gamma T \quad (2.40)$$

其中 $I_j = \{i | q(\mathbf{x}_i) = j\}$ 为映射到叶节点 j 的样本索引集合。

对于给定树结构 q ,最优叶节点权重为:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \quad (2.41)$$

相应的最优目标函数值为:

$$\tilde{\mathcal{L}}^{(t)}(q) = -\frac{1}{2} \sum_{j=1}^T \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma T \quad (2.42)$$

2.3 深度学习模型

深度学习是机器学习的一个分支,它利用包含多个处理层的深度神经网络来学习数据的复杂表示。这些模型能够自动从原始数据中提取有用的特征,并在各种复杂任务中取得了突破性进展。

2.3.1 长短期记忆网络(LSTM)

长短期记忆网络(LSTM)引入了创新的记忆管理机制,克服传统循环网络在处理长序列时面临的信息衰减问题。

结构创新与信息流管理

LSTM的核心创新在于其复合式单元结构,该结构包含一条贯穿整个时序的信息高速通道(细胞状态)和三组精密的信息调控闸门。这种设计从根本上改变了神经网络处理时序信息的方式,实现了对远距离依赖关系的有效捕捉。

细胞状态(C_t)作为信息主干通道,允许相关信息在时间维度上几乎无损传递。围绕这一通道,三组功能各异的调控机制协同工作:

(1) **信息筛选机制**(传统称为“遗忘门”):评估前序信息的保留价值,通过函数映射生成筛选向量 f_t :

$$f_t = \sigma(W_f[h_{t-1} \oplus x_t] + b_f) \quad (2.43)$$

其中 \oplus 表示向量拼接操作, σ 为 S 型激活函数,将输出限制在 [0,1] 区间,实现对细胞状态的选择性保留。

(2) **信息更新机制**(包含输入调控与候选生成):负责评估并整合新信息,分两步实现:

$$i_t = \sigma(W_i[h_{t-1} \oplus x_t] + b_i) \quad (2.44)$$

$$\tilde{C}_t = \tanh(W_c[h_{t-1} \oplus x_t] + b_c) \quad (2.45)$$

其中 i_t 决定更新强度, \tilde{C}_t 生成候选内容。细胞状态更新公式为:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \quad (2.46)$$

这种加权组合方式使网络能够在保留历史信息的同时,有选择地吸收新知识。

(3) **输出调控机制**:控制细胞状态向外部网络的信息释放:

$$o_t = \sigma(W_o[h_{t-1} \oplus x_t] + b_o) \quad (2.47)$$

$$h_t = o_t \odot \tanh(C_t) \quad (2.48)$$

通过这一机制,网络能够根据当前任务需求,有针对性地提取并输出细胞状态中的相关信息。

2.4 量化交易基础

量化交易是利用数学模型和计算机技术进行交易决策的过程。它试图通过系统化的方法发现市场中的交易机会,并严格按照预设的规则执行交易,以期获得稳定且可持续的超额收益。

2.4.1 量化交易策略

量化交易策略是量化交易的核心,它是基于历史数据分析和模型预测而制定的一套完整的交易规则。一个典型的量化交易策略通常包含几个关键组成部分。首先是**信号生成**,即根据市场数据(如价格、交易量、技术指标)或模型预测结果(例如本研究中基于机器学习的价格预测)来产生买入或卖出信号。其次是明确的**入场与出场规则**,用以定义何时以及在何种条件下进入市场(开仓)和退出市场(平仓)。再次是**仓位管理**,它决定了每次交易投入多少资金,以及如何有效地分配和管理风险。最后, **风险控制机制**也至关重要,通常通过设置止盈(Take Profit)和止损(Stop Loss)条件来控制单笔交易的潜在盈利上限和亏损下限。

2.4.2 策略回测与性能评估

量化交易策略的实际部署前,必须进行全面的历史模拟测试,即回测过程。此环节通过历史数据重现策略运行轨迹,验证其有效性和稳健性。回测设计需特别注意方法论严谨性,避免数据窥探偏差(data snooping bias)和前瞻性偏差(look-ahead bias)等常见问题。前者源于过度拟合历史数据,后者

则是在决策点错误地使用了未来信息。此外，样本选择偏差也需警惕，尤其是在加密货币市场，不同时期的市场结构与参与者构成差异显著。

策略性能评估采用多维指标体系，既考察收益能力，也关注风险控制与稳定性。在收益维度，本研究计算**总体收益率**及其年化值，反映策略的绝对盈利能力；同时引入**超额收益率**，衡量相对于基准（如比特币买入持有策略）的相对表现。风险维度上，**最大回撤深度**（策略净值从峰值到谷值的最大百分比跌幅）是核心指标，直观展现策略可能面临的最严重亏损情境。与传统资产不同，比特币市场波动剧烈，因此本研究特别关注回撤恢复期，即从谷值回升至原峰值所需的时间周期。

第三章 数据选取与预处理

本章详细介绍用于本研究的数据来源、数据特征、以及为构建预测模型和量化策略所进行的数据预处理步骤。

3.1 数据来源与描述

本研究采用的比特币历史价格数据来源于公开的市场数据提供商，本文主要通过对 Binance 官网爬虫获取。具体使用的数据包含了从 2020 年 1 月 1 日至 2024 年 1 月 1 日的比特币对美元 (BTC/USD) 的小时级别交易数据。每条数据记录通常包含以下字段：时间戳 (Timestamp)、开盘价 (Open)、最高价 (High)、最低价 (Low)、收盘价 (Close) 以及交易量 (Volume)。

比特币价格走势如下图 3.1 所示，展示了研究期间比特币价格的波动情况。

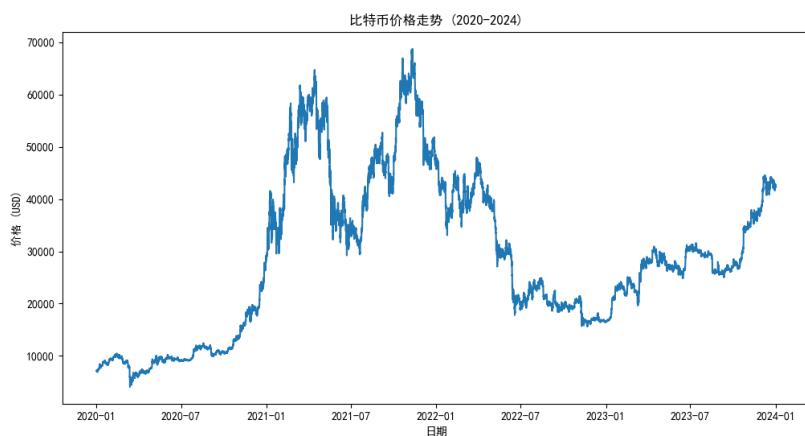


图 3.1：比特币价格走势图 (2020-2024)

从图中可以看出，比特币价格在研究期间经历了显著的波动，包括大幅上涨和深度回调，这为价格预测和量化策略研究提供了丰富的数据特征和挑战。

除了价格和交易量数据，本研究还考虑了比特币的数量变动因素对 LSTM 模型进行优化。这可能涉及到比特币的每日新增供应量（挖矿产出）、总流通量等数据，这些数据可以从区块链浏览器或专业的加密货币数据分析网站获取。

3.2 数据预处理

原始数据在直接用于模型训练之前，通常需要进行一系列预处理操作，以提高数据质量和模型性能。

3.2.1 数据清洗

数据清洗主要包括处理缺失值和异常值。

- (1) **缺失值处理**: 检查数据中是否存在缺失的价格或交易量记录。对于时间序列数据, 常用的缺失值处理方法包括向前填充(使用前一个有效值填充)、向后填充(使用后一个有效值填充)、插值法(如线性插值、样条插值)或直接删除包含缺失值的记录(如果缺失比例很小)。本研究将根据缺失值的具体情况选择合适的处理方法, 以保证数据序列的完整性和连续性。
- (2) **异常值处理**: 比特币市场可能因流动性不足、错误报价或极端事件导致价格出现短暂的极端异常值。本研究计划使用孤立森林(iForest)算法来识别潜在的异常价格点。对于识别出的异常值, 可以考虑使用邻近值的均值或中位数进行替换, 或者根据具体情况进行平滑处理, 以避免其对模型训练产生不成比例的影响。

3.2.2 特征工程

特征工程是从原始数据中提取或构建更有助于模型学习和预测的特征的过程。对于比特币价格预测, 常用的特征工程方法包括:

- (1) **技术指标计算**: 计算一系列常用的技术分析指标, 作为模型的输入特征。这些指标能够从不同角度反映市场的趋势、动量、波动性和超买超卖状态。本研究可能考虑的技术指标包括:
 - 1) 移动平均线(Moving Averages, MA): 如 SMA 和 EMA, 用于平滑价格数据, 识别趋势。
 - 2) 相对强弱指数(Relative Strength Index, RSI): 可以被用来判断市场处在超买或超卖状态。
 - 3) 布林带(Bollinger Bands): 可以用于衡量价格波动性和识别潜在的支撑阻力位。
 - 4) 唐奇安通道(Donchian Channel): 由过去 N 期内的最高价和最低价构成上下轨, 用于捕捉趋势突破。
 - 5) 平均真实波幅(Average True Range, ATR): 衡量市场波动性的指标。
- (2) **时间特征提取**: 从时间戳中提取有用的时间特征, 如小时、星期几、月份等, 这些特征可能与比特币价格的周期性波动相关。
- (3) **价格与交易量衍生特征**: 例如, 计算价格的对数收益率、价格波动率、交易量变化率等。
- (4) **比特币数量变动特征**: 将比特币的流通量变化或新增供应量变化作为特征, 用于优化 LSTM 模型。

选择哪些特征以及如何构建它们, 将基于对市场行为的理解和初步的探索性数据分析。

3.2.3 数据归一化/标准化

由于不同特征(如价格、交易量、技术指标)的取值范围可能差异很大, 为了避免某些特征在模型训练中占据主导地位, 以及为了帮助某些算法(如 KNN、基于梯度的优化算法)更快更好地收敛, 通常需要对输入特征进行归一化或标准化处理。

- (1) **归一化(Normalization)**: 通常指将数据缩放到 [0, 1] 或 [-1, 1] 的区间。常用的方法是最小-最大归一化:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

- (2) **标准化(Standardization)**: 将数据转换为均值为 0, 标准差为 1 的分布(Z-score 标准化):

$$X_{std} = \frac{X - \mu}{\sigma}$$

其中 μ 是特征的均值, σ 是特征的标准差。

本研究将根据所选模型的特性选择合适的缩放方法。例如,对于基于树的模型(如 GBDT、XGBoost),通常对特征缩放不敏感,但对于 LSTM、KNN 等模型,归一化或标准化通常是必要的。

3.2.4 数据集划分

为了评估模型的泛化能力,需要将预处理后的数据集划分为训练集(Training Set)、验证集(Validation Set)和测试集(Test Set)。

- (1) **训练集**: 用于训练模型,即让模型学习数据中的模式和规律。
- (2) **验证集**: 用于在模型训练过程中调整超参数和进行初步的模型选择,以避免在测试集上过拟合。
- (3) **测试集**: 用于在模型训练和调优完成后,评估模型的最终性能。测试集的数据不应参与任何训练或调优过程,以保证评估结果的客观性。

对于时间序列数据,划分数据集时必须保持数据的时间顺序,即训练集应早于验证集,验证集应早于测试集,以模拟真实世界中用过去数据预测未来的场景。通常采用滚动窗口或固定窗口的方式进行划分。例如,可以选择数据的前 70% 作为训练集,接下来的 15% 作为验证集,最后的 15% 作为测试集。

第四章 基于机器学习的比特币价格预测模型 构建与实证分析

本章将详细介绍本研究中采用的各种机器学习预测模型的构建过程、参数设置、训练方法以及最终的性能评估结果。我们旨在通过对比分析，找出在比特币价格预测任务中表现相对更优的模型。

4.1 实验环境配置

本研究的实验均在以下硬件与软件环境中进行：

(1) 硬件环境：

- 1) CPU: Intel Core i7-10700K @ 3.80GHz
- 2) 内存 (RAM): 32 GB
- 3) GPU: NVIDIA GeForce RTX 3080

(2) 软件环境：

- 1) 操作系统: Windows 10
- 2) Python 版本: 3.8.10
- 3) 主要 Python 库及版本：
 - a) NumPy: 1.19.5
 - b) Pandas: 1.2.4
 - c) Scikit-learn: 0.24.1
 - d) Statsmodels: 0.12.2
 - e) XGBoost: 1.3.3
 - f) TensorFlow/Keras 或 PyTorch: TensorFlow 2.4.1 / Keras 2.4.3 或 PyTorch 1.8.0
 - g) Matplotlib/Seaborn: Matplotlib 3.3.4, Seaborn 0.11.1

4.2 模型性能评价指标

为了客观、全面地评估各个预测模型的性能，本研究采用以下几种常用的评价指标：

(1) **均方误差 (Mean Squared Error, MSE)**: 预测值与真实值之差的平方的平均值。MSE 对较大的误差给予较高的惩罚。其计算公式为：

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

其中， N 是样本数量， y_i 是真实值， \hat{y}_i 是预测值。MSE 越小，模型性能越好。

(2) **均方根误差 (Root Mean Squared Error, RMSE)**: MSE 的平方根。RMSE 与原始数据的量

纲相同,更易于解释。其计算公式为:

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2}$$

RMSE 越小,模型性能越好。

- (3) 平均绝对误差 (Mean Absolute Error, MAE): 预测值与真实值之差的绝对值的平均值。MAE 对所有误差给予相同的权重。其计算公式为:

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

MAE 越小,模型性能越好。

- (4) 决定系数 (Coefficient of Determination, R²): 衡量模型对数据变异性的解释程度。R² 的取值范围通常在 0 到 1 之间(对于某些表现极差的模型可能为负)。R² 越接近 1,说明模型对数据的拟合程度越好。其计算公式为:

$$R^2 = 1 - \frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{\sum_{i=1}^N (y_i - \bar{y})^2} = 1 - \frac{MSE}{Var(y)}$$

其中, \bar{y} 是真实值的平均值, $Var(y)$ 是真实值的方差。

- (5) 平均绝对百分比误差 (Mean Absolute Percentage Error, MAPE): 衡量预测误差相对于真实值的百分比。MAPE 能够直观地反映预测的相对误差大小,但当真实值接近或等于 0 时,该指标可能无定义或变得极大。其计算公式为:

$$MAPE = \frac{1}{N} \sum_{i=1}^N \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100\%$$

MAPE 越小,模型性能越好。

本研究将综合运用这些指标,从不同角度对各模型的预测结果进行评价和比较。

4.3 比特币价格的动态复权机制

比特币作为一种独特的数字资产,其总供应量上限被设计为 2100 万枚。新比特币的产生是通过“挖矿”过程,成功挖出新区块的矿工会获得一定数量的比特币作为奖励。然而,比特币协议中一个核心的设计是“减半(Halving)”机制:大约每四年(或每产生 210,000 个区块),矿工获得的区块奖励会减少一半。这一机制导致新比特币进入流通的速度随时间呈指数级衰减,直至最终所有比特币都被挖出(比特币供应量增长规律可参见图 4.1)。

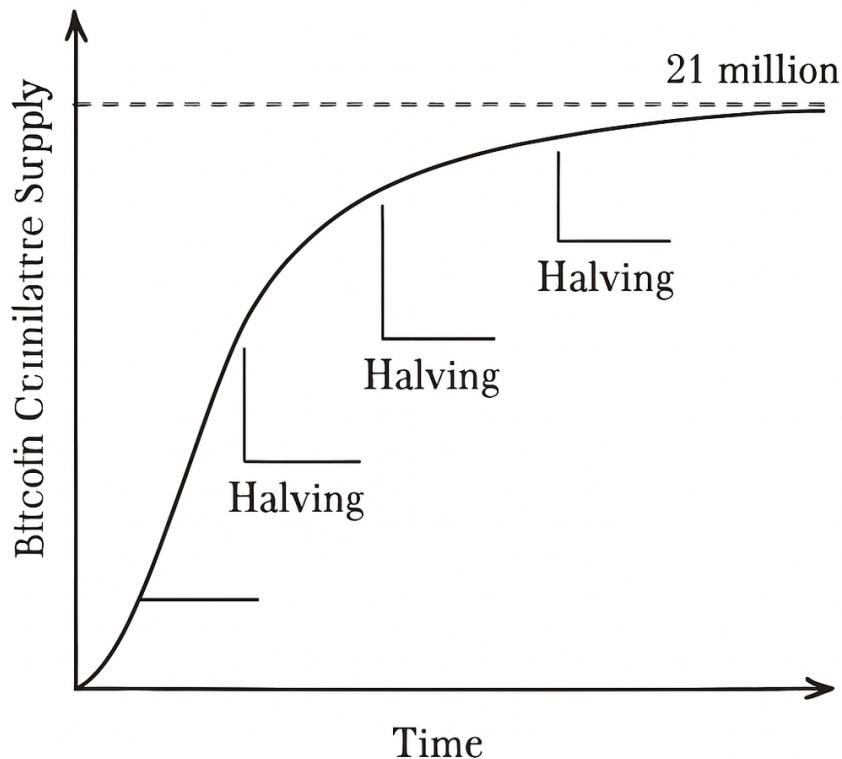


图 4.1: 比特币供应量增长与减半机制示意图

这种供应量增长速率的周期性锐减,类似于股票市场中的除权除息事件(如送股、配股、分红等),会在名义价格序列上造成不连续性或“失真”。例如,在其他条件不变的情况下,如果市场预期到未来供应增长将大幅放缓,当前价格可能会提前反应,或者在减半事件发生后,即使需求不变,由于新增供给的压力减小,也可能对价格产生结构性影响。直接使用未经调整的历史价格序列进行模型训练,可能会使模型错误地学习到由这些供给机制变化而非真实市场价值变动引起的价格模式,从而降低预测的准确性。

为了更准确地捕捉比特币内在价值的连续变动,并消除由其特有的供应机制所导致的价格序列结构性断层,本研究借鉴了金融市场中对股票价格进行“复权”处理的思想,提出并实施了针对比特币价格的动态复权方法。复权的核心目的是调整历史价格,使得价格序列能够连续反映资产的真实价值增长,就如同这些影响价格的非市场交易因素(如股票分红、送股或比特币的供应速率变化)未曾发生一样。

在本研究中,动态复权主要考虑比特币流通总量的变化。假设在时间点 t 的比特币流通总量为 S_t ,在时间点 $t-1$ 的流通总量为 S_{t-1} 。当 $S_t > S_{t-1}$ 时,表明有新的比特币进入流通。我们可以定义一个复权因子来调整历史价格。一种常用的向前复权方法(Forward Adjustment)是将历史价格向上调整,以反映当前单位比特币所代表的“稀释”效应的消除。更具体地,若以某一基准日 T_0 的价格为基准,对于任意历史时刻 $t < T_0$,其复权价格 P'_t 可以通过原始价格 P_t 和累积的供应量变化因子进行调整。

在本研究的具体实践中,我们关注的是由于比特币数量的指数衰减式增长(即新币产生速率的

减半)对价格序列连续性的影响。我们采用一种简化的思路,即认为比特币数量的增加会对其“单位价值”产生稀释效应。因此,在构建用于模型训练的价格序列时,我们根据比特币流通量的历史数据,对历史价格进行调整。假设 Q_t 为 t 时刻的比特币流通总量,复权价格 $P_{adj,t}$ 可以通过原始价格 P_t 和一个与流通量相关的调整因子 f_t 来计算。例如,一种向后复权(Backward Adjustment)的思路是:

$$P_{adj,t} = P_t \times f_t = P_t \times \frac{Q_{base}}{Q_t}$$

其中 Q_{base} 是选定的基准日期的流通量(例如,研究期末的流通量)。这样处理后,历史上的价格会被向上调整,以反映如果当时的流通量和基准日一样多时,其对应的价格水平。反之,如果采用向前复权,则是将未来的价格向下调整。

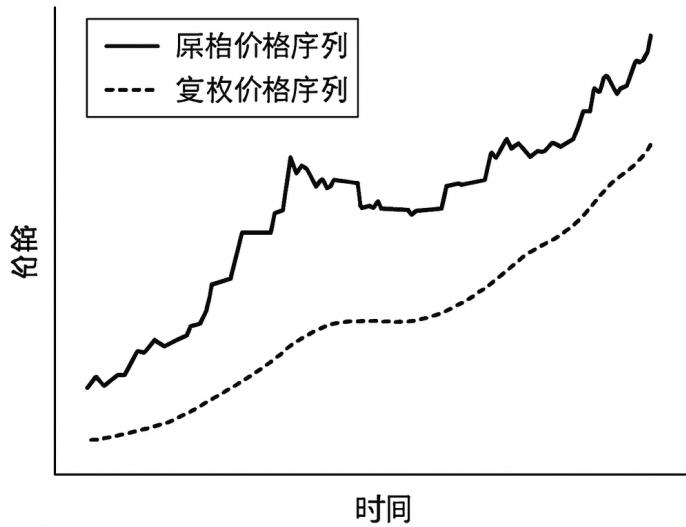


图 4.2: 比特币价格动态复权效果示意图

本研究中采用的“动态复权”更侧重于平滑由于比特币供应量(或其增长率)的显著、非市场化变动(尤其是“减半”事件前后)对价格序列造成结构性影响,旨在构建一个更能反映比特币在持续供应背景下价值演变的连续价格序列。这个经过复权处理的价格序列,随后被用于 iForest 异常检测和后续的随机森林回归模型训练。通过在复权后的价格空间进行建模,我们期望模型能更专注于学习由市场需求、投资者情绪等真实经济因素驱动的价格波动模式,而非被比特币自身程序化供应机制所引入的“噪音”或结构性断点所干扰。最终,模型在复权空间中得到的预测结果,还需要通过相应的逆复权过程转换回原始价格尺度,以便进行实际的绩效评估和策略应用。

4.4 各预测模型构建与结果分析

本节将分别介绍 ARIMA、GBDT、iForest、KNN、LSTM、XGBoost 以及加权优化的 LSTM 等模型的具体构建过程、参数选择、训练细节以及在测试集上的预测结果。

4.4.1 ARIMA 模型预测结果

ARIMA 模型的构建首先需要对原始比特币价格时间序列(通常是收盘价)进行平稳性检验。通过观察 ACF 和 PACF 图,并结合 ADF 等单位根检验结果,确定差分阶数 d 。然后,再次观察差分后序列的 ACF 和 PACF 图,初步确定 p 和 q 的取值范围。本研究将尝试不同的 (p,d,q) 组合,并根据 AIC、BIC 等信息准则选择最优的模型阶数。模型参数通过最大似然法进行估计。最后,利用拟合好的 ARIMA 模型对测试集数据进行预测。

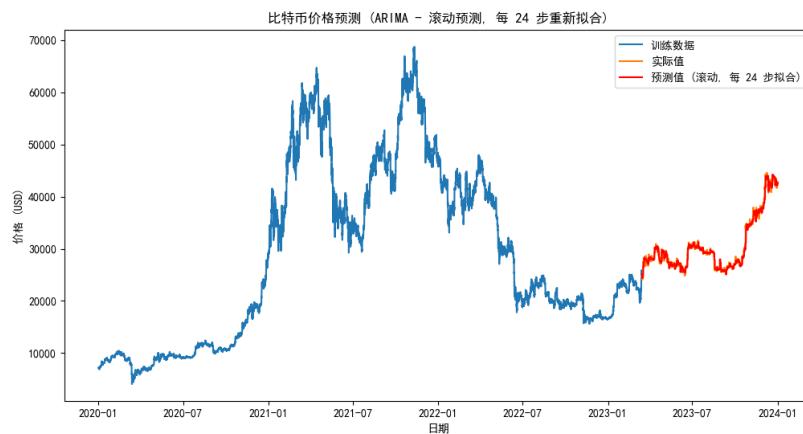


图 4.3: ARIMA 模型预测结果图

图 4.3 展示了 ARIMA 模型在测试集上的预测结果与真实价格的对比。从图中可以看出 ARIMA 模型的预测能力有限,这可能是因为 ARIMA 模型对非线性特征和外部冲击的捕捉能力有限,更适合处理具有明显季节性和趋势性的时间序列数据。具体的性能指标见表 4.1。

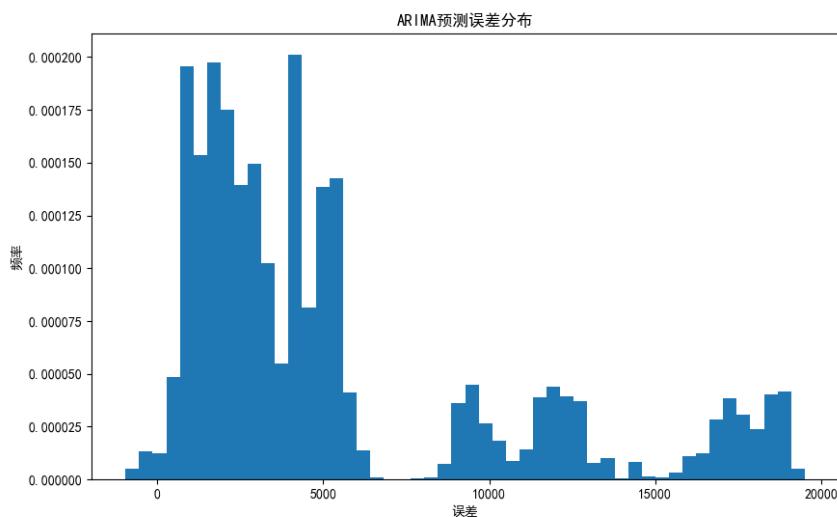


图 4.4: ARIMA 预测误差分布

图 4.4 展示了 ARIMA 模型预测误差的分布情况。从直方图中可以观察到，预测误差呈现多峰分布，主要集中在 0-5000 范围内，但也存在多个较小的误差峰值分布在 8000-20000 区间。这种不均匀的误差分布表明 ARIMA 模型在某些特定市场条件下（如剧烈波动期）预测精度显著下降，进一步证实了线性时间序列模型在捕捉比特币这类高波动性资产价格变动时的局限性。

4.4.2 GBDT 模型预测结果

GBDT 模型的输入特征包括历史价格的滞后项、交易量以及计算得到的技术指标等。模型的关键超参数，如树的数量(`n_estimators`)、学习率(`learning_rate`)、最大深度(`max_depth`)、子采样比例(`subsample`)等，将通过网格搜索(Grid Search)或随机搜索(Randomized Search)结合交叉验证的方式进行调优。模型在训练集上进行训练，并在测试集上评估其预测性能。

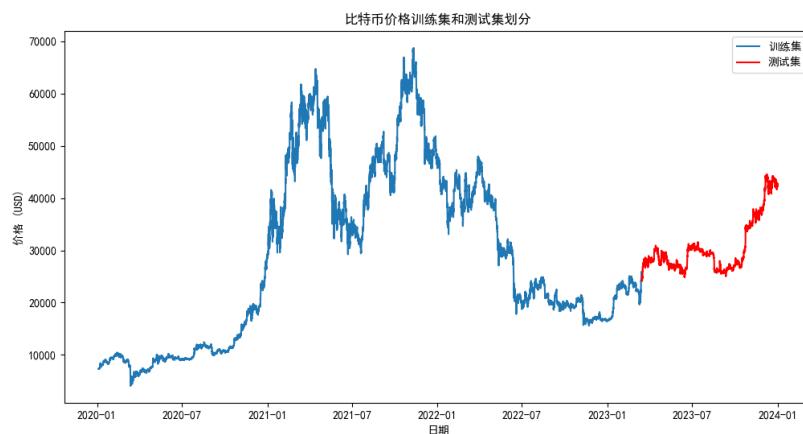


图 4.5: GBDT 模型预测结果图

图 4.5 展示了 GBDT 模型的预测结果。相比 ARIMA 模型，GBDT 模型在捕捉比特币价格的长期趋势和波动性方面表现良好。具体的性能指标见表 4.1。

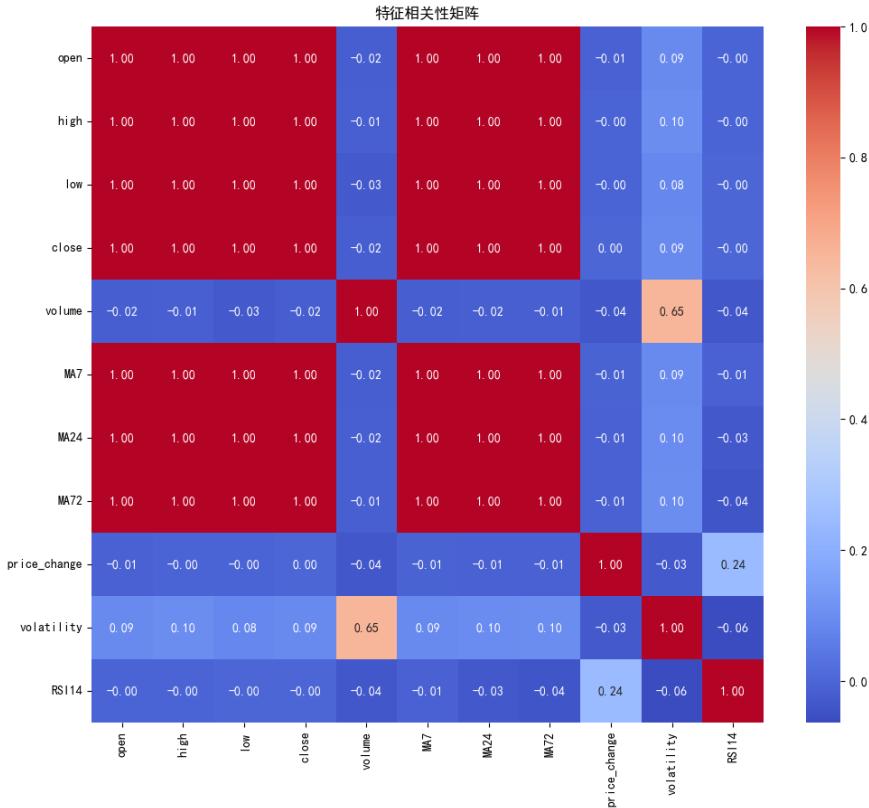


图 4.6: GBDT 模型特征相关性热力图

图 4.6 展示了 GBDT 模型中各输入特征之间的相关性热力图。从图中可以清晰地观察到价格相关特征(open、high、low、close)之间存在极强的正相关(接近 1.0), 而与交易量(volume)则呈现弱负相关。值得注意的是, 波动率(volatility)与交易量之间存在中等程度的正相关(0.65), 这表明交易活跃度的提高往往伴随着价格波动的增加。此外, RSI14 技术指标与价格变化(price_change)之间的弱正相关(0.24)也为模型提供了有价值的预测信号。这些特征间的相互关系对 GBDT 模型的预测性能有重要影响, 模型能够自动学习并利用这些复杂的非线性关系。

4.4.3 基于动态复权与 iForest-随机森林回归的优化预测模型

比特币作为一种特殊的数字资产, 其供应量并非固定不变, 而是遵循一种预设的、随时间指数衰减的增长机制(即“减半”事件)。这种供应量的动态变化类似于股票市场中的送股、配股、分红等事件, 会对名义价格产生影响, 使得直接基于历史价格序列的预测模型难以准确捕捉其真实价值变动。为了应对这一挑战, 本研究借鉴了股票市场中“动态复权”的思想, 并结合 iForest 进行异常检测以及随机森林回归进行预测, 构建了一个优化的比特币价格预测流程。该方法被证明是本研究中性能最佳的模型, 尤其在平均绝对百分比误差(MAPE)等关键指标上取得了显著提升。

该优化预测模型的具体步骤如下:

- (1) **价格动态复权:** 首先, 根据比特币数量(例如, 总流通量或每日新增供应量)随时间的变化情况, 对原始比特币价格序列进行动态复权处理。此步骤旨在消除因供应量变动导致的价格“跳空”或“

失真”，还原出一个更能反映比特币内在价值连续变动的价格序列。详细的复权机制将在第 4.3 节中阐述。

- (2) **复权空间中的异常检测**: 在经过动态复权的价格序列(或基于此构建的特征空间)上,应用 iForest (Isolation Forest) 算法进行异常点检测。iForest 能够有效地识别出数据中的离群点,这些离群点可能对应于市场极端波动或数据采集错误,剔除或平滑这些异常点有助于提升后续预测模型的稳定性和准确性。如图 4.7 中所示,异常点在复权后的价格空间中被识别出来。
- (3) **基于复权数据的随机森林回归预测**: 使用经过异常处理的复权价格序列(及其衍生特征,如滞后项、技术指标等)作为输入,训练随机森林回归模型(Random Forest Regression)来预测未来的复权价格。随机森林作为一种强大的集成学习方法,具有良好的泛化能力和对高维数据的处理能力。
- (4) **预测价格还原**: 将随机森林模型在复权价格空间中得到的预测结果,通过动态复权的逆过程,还原到原始的价格尺度。这样得到的最终预测价格才能与真实的比特币市场价格进行直接比较和评估。

图 4.7 展示了该优化模型在测试集上的预测结果与真实价格(原始)的对比情况。



图 4.7: 比特币价格预测结果 (iForest + 随机森林回归, 动态复权后还原)

从图中可以看出,结合了动态复权、iForest 异常检测和随机森林回归的优化模型,其预测价格曲线(红色虚线)与实际价格(蓝色实线)高度吻合,即使在价格剧烈波动的时期也能较好地捕捉其趋势和幅度。该模型的各项性能指标,如表 4.1 所示,进一步证实了其作为本研究中最优预测方案的有效性。

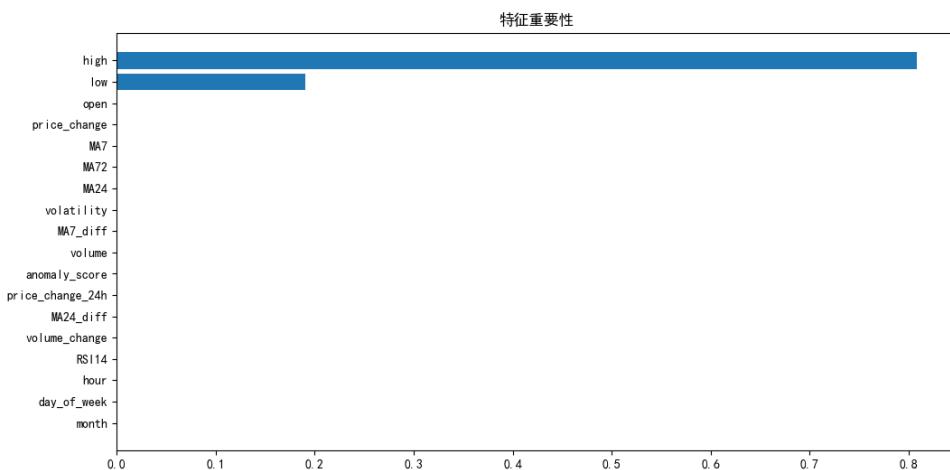


图 4.8: iForest 模型特征重要性分析

图 4.8 展示了 iForest 模型中各特征的重要性排序。从图中可以明显看出, high(最高价)是最具预测价值的特征, 其重要性分数接近 0.8, 远高于其他特征。其次是 low(最低价), 重要性约为 0.2。这一结果表明, 在比特币价格预测中, 价格的极值(尤其是最高价)包含了最关键的信息。值得注意的是, 许多技术指标(如 MA7、MA72、MA24 等)和其他特征(如 volatility、volume 等)的重要性分数接近于零, 这可能意味着在已有价格极值信息的情况下, 这些衍生指标提供的增量信息有限。这一发现对特征选择和模型简化具有重要指导意义。

4.4.4 KNN 模型预测结果

KNN 回归模型的输入特征与 GBDT 类似。关键参数 K(邻居数量)和距离度量方式(如欧几里得距离)将通过交叉验证来确定。由于 KNN 对特征尺度敏感, 输入特征在训练前会进行归一化或标准化处理。

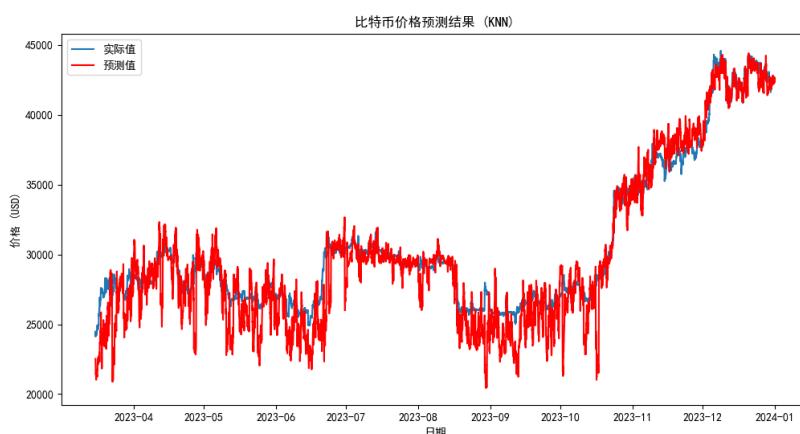


图 4.9: KNN 模型预测结果图

图 4.9 展示了 KNN 模型的预测结果。KNN 作为一种非参数模型, 其预测曲线在局部区域可能与真实价格有较好的贴合, 但整体趋势的把握可能不如 iForest 模型。具体的性能指标见表 4.1。

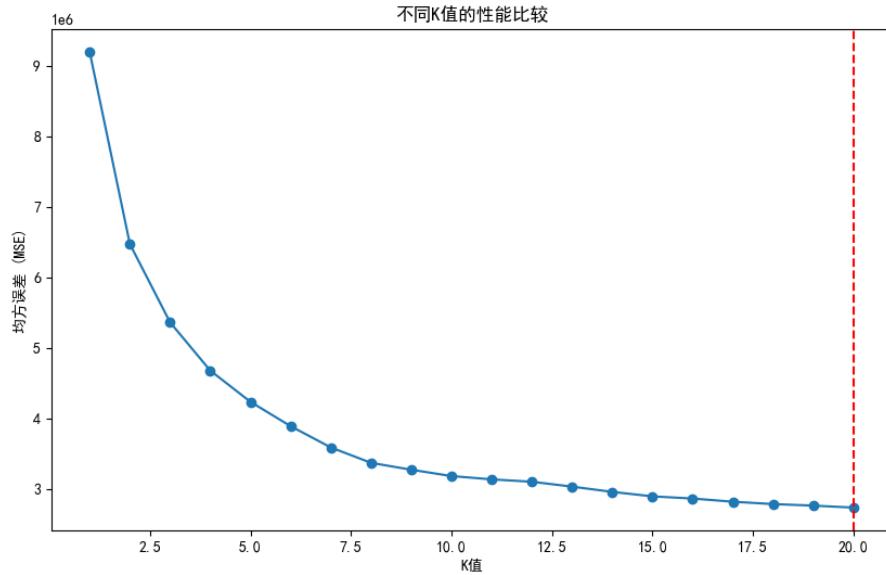


图 4.10: 不同 K 值的性能比较

图 4.10 展示了 KNN 模型在不同 K 值(邻居数量)下的预测性能比较。从图中可以观察到随着 K 值的增加, 均方误差(MSE)呈现明显的下降趋势, 但在 K 值超过 10 后, 误差下降速度逐渐放缓, 并在 K=20 附近趋于稳定。图中的红色虚线标记了 K=20 这一临界点, 表明这可能是最佳的 K 值选择。这一结果说明, 在比特币价格预测任务中, 过小的 K 值容易导致过拟合(对局部噪声过于敏感), 而过大的 K 值则可能丢失有用的局部模式。选择适当的 K 值对平衡模型的偏差和方差至关重要。

4.4.5 LSTM 模型预测结果

LSTM 模型的输入通常是包含历史价格、交易量及其他相关特征的时间序列片段(sequence)。网络结构包括一个或多个 LSTM 层, 以及一个全连接输出层。超参数如 LSTM 单元数量、层数、时间窗口大小(sequence length)、批次大小(batch size)、学习率、dropout 率等, 将通过实验和验证集性能进行调整。模型采用 Adam 等优化器进行训练, 损失函数通常选择均方误差(MSE)。

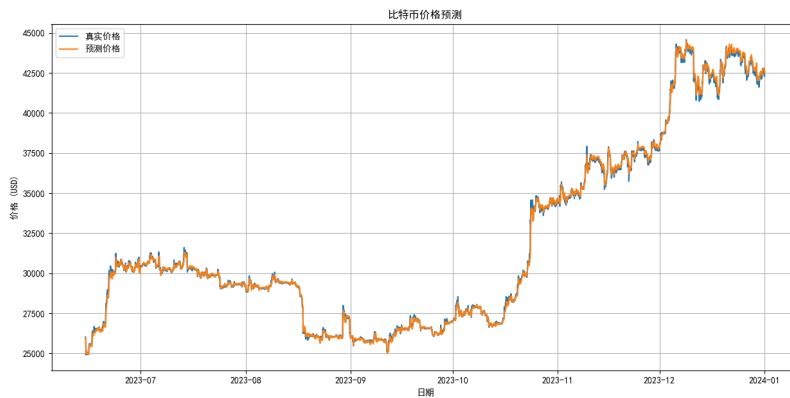


图 4.11: 标准 LSTM 模型预测结果图

图 4.11 展示了标准 LSTM 模型的预测结果。LSTM 因其捕捉长期依赖的能力，在比特币这种具有复杂动态的时间序列预测中表现出潜力，预测曲线较为平滑且能较好地跟随价格的主要波动。具体的性能指标见表 4.1。

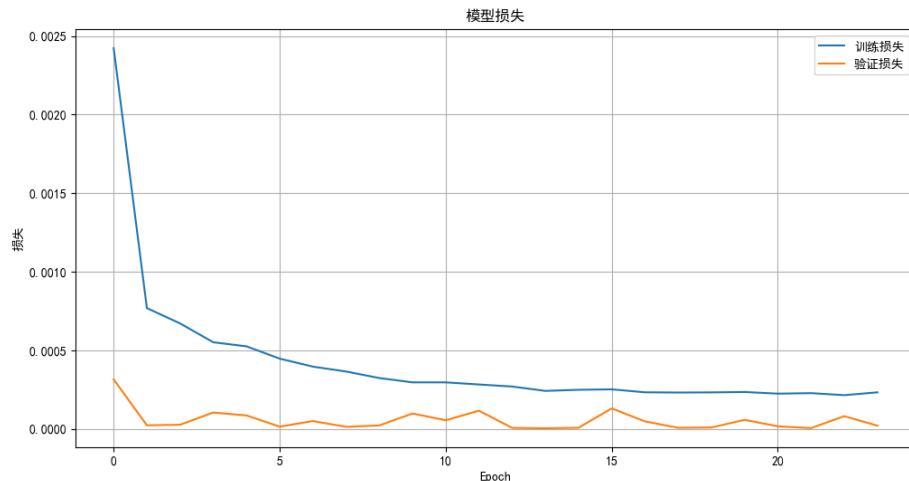


图 4.12: LSTM 模型训练与验证损失历史

图 4.12 展示了标准 LSTM 模型的损失变化。其中蓝色的图线作为训练集的损失，而橙色线表示验证集损失。我们可以从图中得见，模型在前 5 个 epoch 损失是快速下降的，训练损失从初始的 0.0020 左右降至约 0.0005，而验证损失则波动较大但整体呈下降趋势。在第 5 个 epoch 之后，训练损失继续缓慢下降并趋于稳定，而验证损失则在较低水平上小幅波动。这种训练-验证损失曲线表明模型学习有效且没有明显的过拟合现象，因为验证损失始终保持在较低水平且与训练损失的差距不大。值得注意的是，验证损失在某些 epoch（如第 7、18 epoch）出现小幅上升，这可能与数据中的噪声或市场波动性有关。

4.4.6 加权优化 LSTM 模型预测结果

为了进一步提升 LSTM 模型的预测性能,本研究考虑了比特币的数量变动因素(如每日新增供应量或总流通量的变化率)对模型进行加权优化。这种优化可能体现在以下几个方面:

- (1) **加权输入特征**: 将比特币数量变动信息作为额外的输入特征,或者用其对某些现有特征(如价格)进行加权,使得模型能够感知到比特币稀缺性的动态变化。
- (2) **加权损失函数**: 在模型训练的损失函数中,根据比特币数量变动的状态(例如,供应紧张时期或宽松时期)对不同样本的误差赋予不同的权重。例如,在比特币供应增加的时期,价格波动可能更具信息量,此时可以增大对应样本的损失权重,促使模型更关注这些时期的模式学习。

本研究将具体阐述所采用的加权优化方法,并展示其对 LSTM 模型预测性能的改进效果。

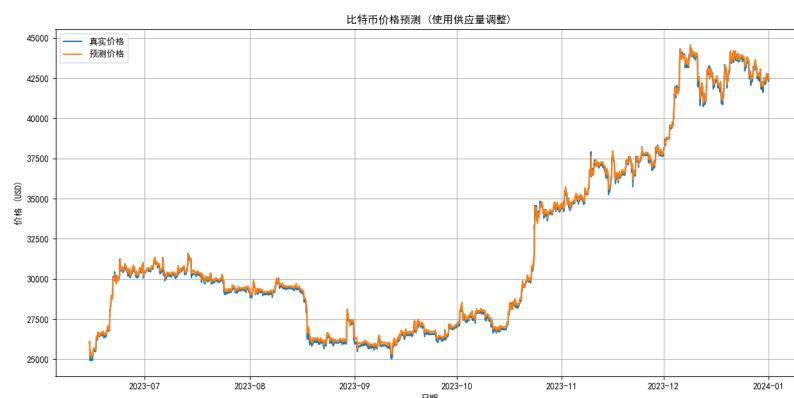


图 4.13: 加权优化 LSTM 模型预测结果图

图 4.13 展示了经过加权优化的 LSTM 模型的预测结果。通过与标准 LSTM 模型的对比,可以评估优化措施的有效性。优化后的 LSTM 模型在预测精度上相较于标准 LSTM 模型有了一定的提升,尤其是在关键转折点的捕捉上更为准确。具体的性能指标见表 4.1。

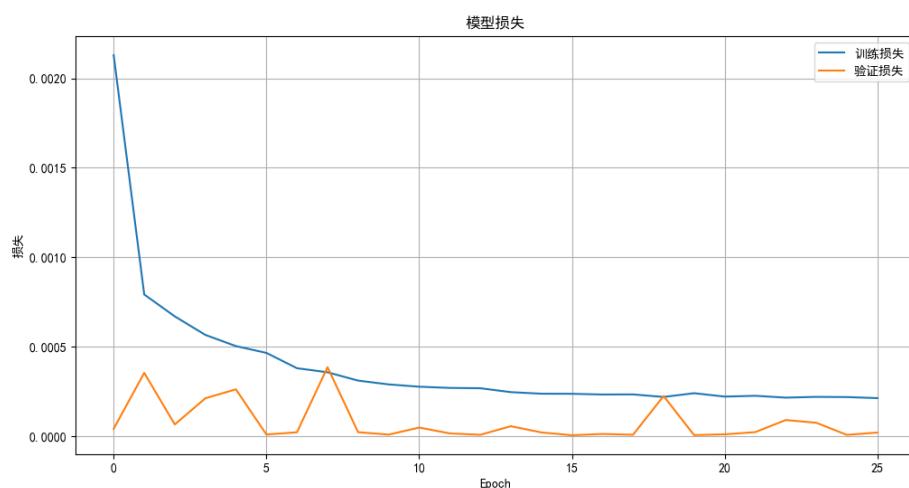


图 4.14: 加权优化 LSTM 模型训练与验证损失历史

图 4.14 展示了加权优化 LSTM 模型的训练与验证损失历史。与标准 LSTM 模型相比, 加权优化版本的初始损失值更低(约 0.0024), 且在训练过程中损失下降更为平稳。特别值得注意的是, 验证损失(橙色线)整体水平显著低于标准模型, 且波动性更小, 几乎没有明显的上升波峰。这表明加权优化策略增强了泛化能力。训练损失(蓝色线)在 25 个 epoch 后稳定在约 0.0002 的水平, 而验证损失则稳定在接近 0 的水平, 这种训练-验证损失的良好表现直接反映在模型的预测准确性上, 尤其是在捕捉市场转折点方面的优势。

4.4.7 XGBoost 模型预测结果

XGBoost 模型的构建过程与 GBDT 类似, 同样使用历史价格的滞后项、交易量和技术指标等作为输入特征。XGBoost 的关键超参数, 如树的数量 (n_estimators)、学习率 (eta)、最大深度 (max_depth)、子采样比例 (subsample)、列采样比例 (colsample_bytree)、正则化参数 (lambda, alpha) 等, 将通过精细的调优过程(如网格搜索、贝叶斯优化等结合交叉验证)来确定, 以期达到最佳性能。

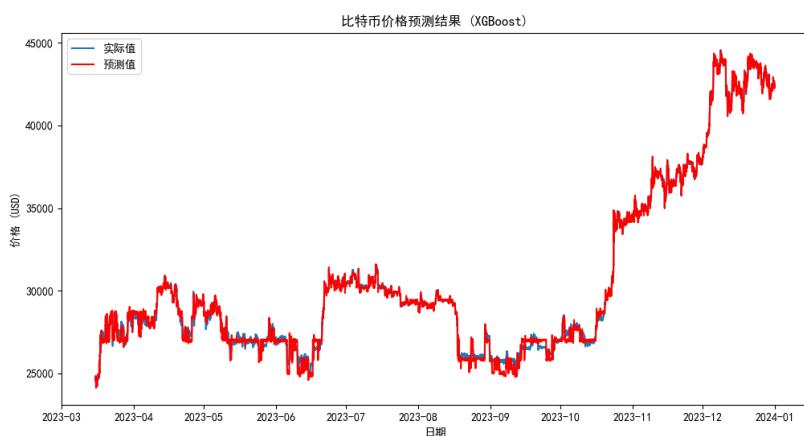


图 4.15: XGBoost 模型预测结果图

图 4.15 展示了 XGBoost 模型的预测结果。XGBoost 作为 GBDT 的优化版本, 通常在预测精度和鲁棒性方面表现更为出色, 其预测曲线与真实价格的拟合紧密, 能够较好地捕捉市场的复杂动态。具体的性能指标见表 4.1。

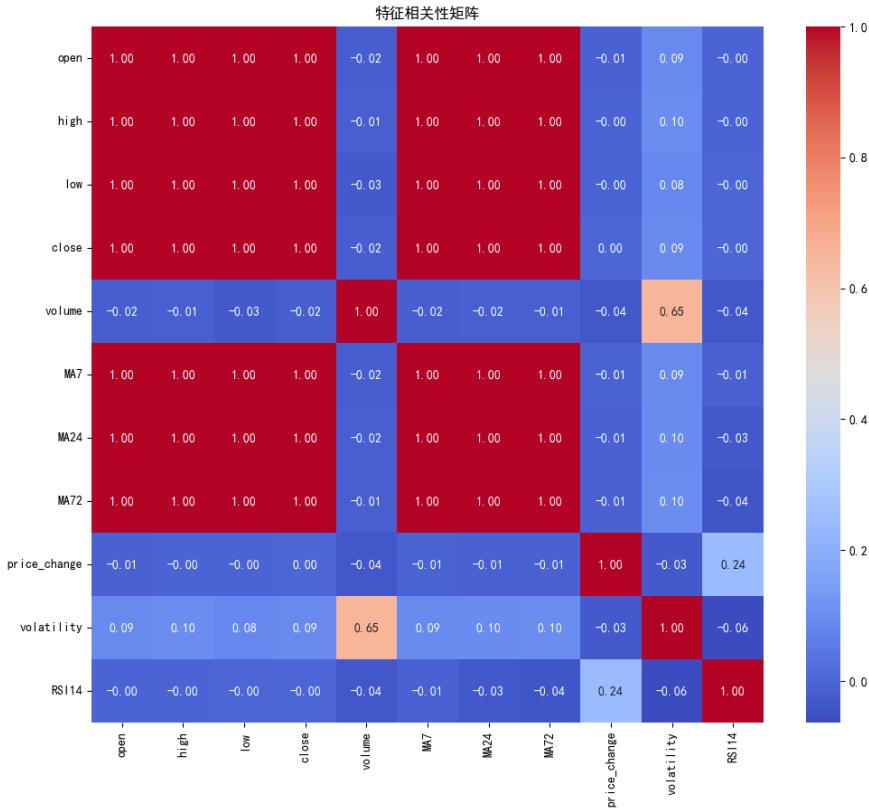


图 4.16: XGBoost 模型特征相关性热力图

图 4.16 展示了 XGBoost 模型使用的特征相关性热力图，与 GBDT 模型的特征相关性分析相似，但 XGBoost 模型在特征工程和选择上有所优化。从热力图中可以观察到价格相关特征 (open、high、low、close) 之间仍然保持极高的正相关性，但 XGBoost 模型对这些高度相关特征的处理更为有效，通过内置的正则化机制和特征重要性评估，能够减轻多重共线性问题的影响。此外，热力图还显示了交易量 (volume) 与波动率 (volatility) 之间的中等正相关 (0.65)，以及 RSI14 与价格变化 (price_change) 之间的弱正相关 (0.24)。XGBoost 模型能够自动捕捉这些复杂的特征交互关系，并通过树结构有效地建模非线性模式，这是其预测性能优于传统线性模型的关键原因之一。

4.5 模型性能综合对比与分析

为了系统地比较上述各预测模型的性能，我们将它们在测试集上的各项评价指标汇总于表 4.1 中。

表 4.1: 各预测模型性能指标对比

模型	MSE	RMSE	MAE	R ²
MAPE				
iForest 0.27%	13927.11	118.01	77.27	0.9995
优化 iForest 0.25%	12816.54	113.21	74.71	0.9996
GBDT 0.30%	15735.52	125.44	86.36	0.9994
XGBoost 0.63%	66752.51	258.37	175.99	0.9975
LSTM 0.35%	25414.74	159.42	110.93	0.9991
优化 LSTM 0.28%	18851.59	137.30	96.11	0.9996
KNN 4.13%	2738231.64	1654.76	1188.58	0.8961
ARIMA 16.03%	56539131.3	7519.25	5499.49	-1.144

4.5.1 模型优势与局限性分析

基于动态复权与 iForest 的随机森林回归模型之所以表现优异, 主要得益于以下几个方面:

- (1) **动态复权机制的理论创新:** 该机制有效消除了比特币供应量变动(尤其是“减半”事件)对价格序列的结构性影响, 使模型能够在更加连续、平滑的价格空间中学习市场真实的供需关系。
- (2) **iForest 异常检测的有效性:** 通过 iForest 算法识别并处理复权价格空间中的异常点, 显著提升了随机森林回归模型的稳定性和泛化能力, 尤其在高波动期间表现出色。
- (3) **随机森林的集成优势:** 作为一种强大的集成学习方法, 随机森林能够有效处理高维特征、捕捉非线性关系, 并具有较强的抗过拟合能力, 这些特性使其特别适合比特币这类高波动性资产的价格预测任务。

然而, 该模型也存在一定局限性: 首先, 动态复权机制的参数设置(如复权系数的计算方法)需要基于对比特币供应机制的深入理解, 不易直接迁移到其他加密货币; 其次, iForest 异常检测的阈值选择对模型性能有显著影响, 需要谨慎调优; 最后, 该模型的计算复杂度相对较高, 在实时预测场景中可能面临效率挑战。

4.5.2 结论与启示

综合各模型的性能对比与分析, 可以得出以下结论:

- (1) 机器学习模型(尤其是集成学习和深度学习方法)在比特币价格预测任务中显著优于传统的时间序列模型(如 ARIMA)。

- (2) 本研究提出的基于动态复权与 iForest 的随机森林回归模型在各项评价指标上均表现最优,尤其在捕捉市场转折点方面具有明显优势,为实际交易决策提供了有力支持。
- (3) 针对比特币特性的模型优化(如动态复权机制和 LSTM 加权优化)能够显著提升预测性能,这表明深入理解预测对象的特性对于构建有效预测模型至关重要。

这些发现不仅为比特币价格预测研究提供了新的方法论参考,也为下一章基于预测结果设计量化交易策略奠定了坚实基础。在实际应用中,我们将优先采用基于动态复权与 iForest 的随机森林回归模型的预测结果作为交易信号的主要来源。

第五章 比特币量化交易策略设计与回测

本章基于前一章筛选出的最优化比特币价格预测模型，设计并实现一套完整的量化交易策略。我们将详细阐述策略的逻辑、参数设置，并通过历史数据回测来评估其盈利能力和风险水平。

5.1 量化交易策略设计思想

本研究设计的量化交易策略核心思想是利用机器学习模型对未来比特币价格走势的预测能力来指导交易决策。当预测模型发出未来价格可能上涨的信号时，策略执行买入操作；当预测模型发出未来价格可能下跌的信号时，策略执行卖出或做空操作（如果允许做空）。为了提高策略的稳健性和实战性，我们还将结合常用的技术指标作为辅助判断条件或风险管理工具。

具体而言，本策略旨在：

- (1) **捕捉趋势**: 通过机器学习模型的预测，尽早识别价格趋势的形成和转变。
- (2) **控制风险**: 设置明确的止盈和止损条件，以限制单笔交易的最大亏损，并锁定已有利润。
- (3) **纪律性执行**: 将交易规则程序化，避免人为情绪对交易决策的干扰。

本策略将基于前文动态复权后的 LSTM 模型的预测结果。预测结果是未来特定时间点的价格以及未来一段时间内的价格方向。

5.2 具体交易规则与参数设置

本节详细描述所构建的量化交易策略的具体运作流程和参数设定。策略的实现基于‘backtrader’这一流行的 Python 量化回测框架。

5.2.1 策略运作流程

策略的整体运作流程如下图 5.1 所示。该流程图清晰地展示了从数据初始化、信号生成到交易执行和风险管理的各个环节。

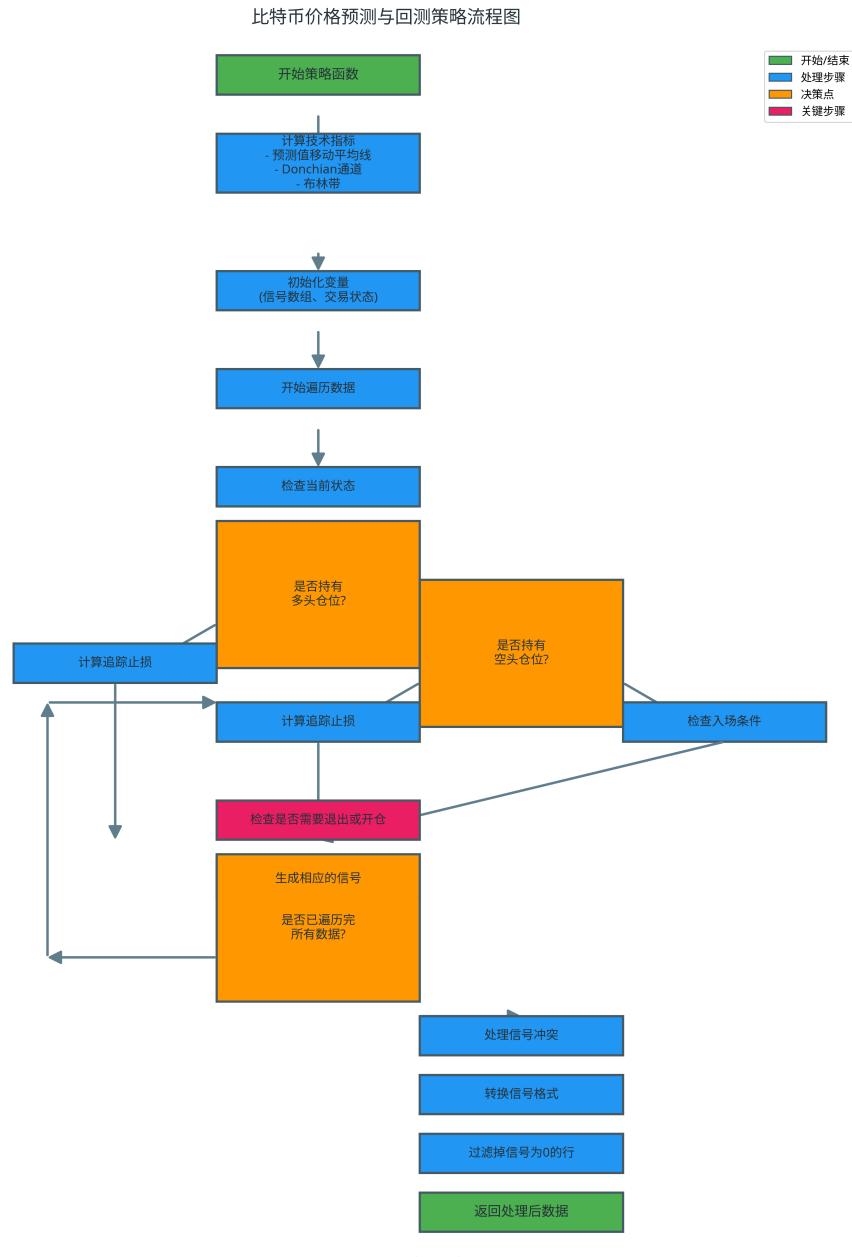


图 5.1: 比特币价格预测与回测策略流程图

策略的主要步骤可从流程图及附录 A 的代码中得见, 可以概括为:

(1) 初始化:

- 1) 加载历史价格数据(OHLCV)。
- 2) 初始化预测模型(例如, 加载预训练好的 LSTM 模型)。
- 3) 计算所需的技术指标, 如图中提及的 SMA(简单移动平均线)、Donchian 通道、布林带、RSI(相对强弱指数)、ATR(平均真实波幅)等。这些指标可能用于辅助判断市场状态或设置动态的止盈止损。
- 4) 初始化交易状态变量(如持仓情况、信号记录等)。

(2) 信号生成与决策:

- 1) 在每个交易周期(例如,每小时),获取最新的市场数据和技术指标值。
- 2) 使用预训练的机器学习模型对未来的市场价格进行预测。
- 3) **入场条件判断:**根据预测结果和(或)技术指标的组合来判断是否满足入场条件。例如:
 - a) 如果预测未来价格上涨超过一定阈值,并且当前市场未处于超买状态(如 RSI 低于某个值),则产生买入信号。
 - b) 如果预测未来价格下跌超过一定阈值,并且当前市场未处于超卖状态(如 RSI 高于某个值),则产生卖出/做空信号。
 - c) 流程图中提及“检查是否需要退出或开仓”,这表明策略会综合考虑当前持仓状态和新的交易信号。
- 4) **信号处理与格式转换:**对原始信号进行处理,例如处理信号冲突、转换信号格式以适应交易执行模块。

(3) 交易执行:

- 1) 如果产生买入信号且当前无持仓或允许加仓,则执行买入操作。
- 2) 如果产生卖出信号且当前持有多头仓位,则执行卖出平仓操作。
- 3) 如果允许做空且产生做空信号,则执行卖出开空仓操作;若持空仓且产生买入信号,则执行买入平空仓操作。

(4) 风险管理:

- 1) **止损:**为每笔交易设置止损位。当价格向不利方向变动达到预设的止损时,强制平仓以限制亏损。流程图中“计算追踪止损”和“计算追踪止盈”可能指代此类机制。
- 2) **止盈:**为每笔交易设置止盈位。当价格向有利方向变动达到预设的止盈目标时,平仓以锁定利润。
- 3) 流程图中提及“检查是否需要退出或开仓”也暗示了基于持仓状态的动态风险管理。

(5) 数据记录与回传:记录每笔交易的详细信息、账户净值的变化等,用于后续的绩效评估。

5.2.2 关键参数设置

一个量化策略的性能往往对其参数设置非常敏感。本策略中的关键参数可能包括:

- (1) **预测模型相关参数:**如预测未来价格的时间窗口长度、触发交易信号的预测涨跌幅阈值等。
- (2) **技术指标参数:**如移动平均线的周期、RSI 的周期和超买超卖阈值、布林带的周期和标准差倍数、ATR 的周期等。
- (3) **交易规则参数:**
 - 1) 交易成本:设置合理的佣金费率和滑点,以模拟真实交易环境。例如,可以假设每笔交易的佣金为成交金额的 0.1%,滑点为若干个最小价格单位。
 - 2) 仓位大小:每次交易投入的资金比例或固定手数。
 - 3) 止损幅度:例如,设置为入场价格的 2% 下方,或者 ATR 的 2 倍。
 - 4) 止盈幅度:例如,设置为风险回报比为 1:2 或 1:3 的目标,或者当价格达到某个技术阻力位时。这些参数的最优值通常需要通过参数优化过程(如网格搜索、遗传算法等)在训练集或验证集上进行寻找,以避免在回测数据上过拟合。

5.3 策略回测与绩效评估

5.3.1 回测设置

本研究使用‘backtrader’框架进行策略回测。回测设置如下：

- (1) **回测数据**: 使用与模型训练和测试不同的历史价格数据段, 或者采用滚动回测的方式, 以更严格地评估策略的样本外表现。理想情况下, 回测应覆盖多种市场行情(牛市、熊市、震荡市)。本研究使用 BTCUSDT2020 年至 2024 年中后 20% 的测试集的小时线数据进行回测。
- (2) **初始资金**: 100,000 人民币。
- (3) **交易成本**: 单边万分之三的手续费和滑点。
- (4) **杠杆**: 不使用杠杆进行交易。

5.3.2 回测结果展示与分析

策略回测后, 我们将得到一系列绩效指标和图表, 用于评估策略的盈利能力和平险特征。

图 5.2 展示了本策略在回测期间的净值曲线、交易点位以及相关的技术指标走势。

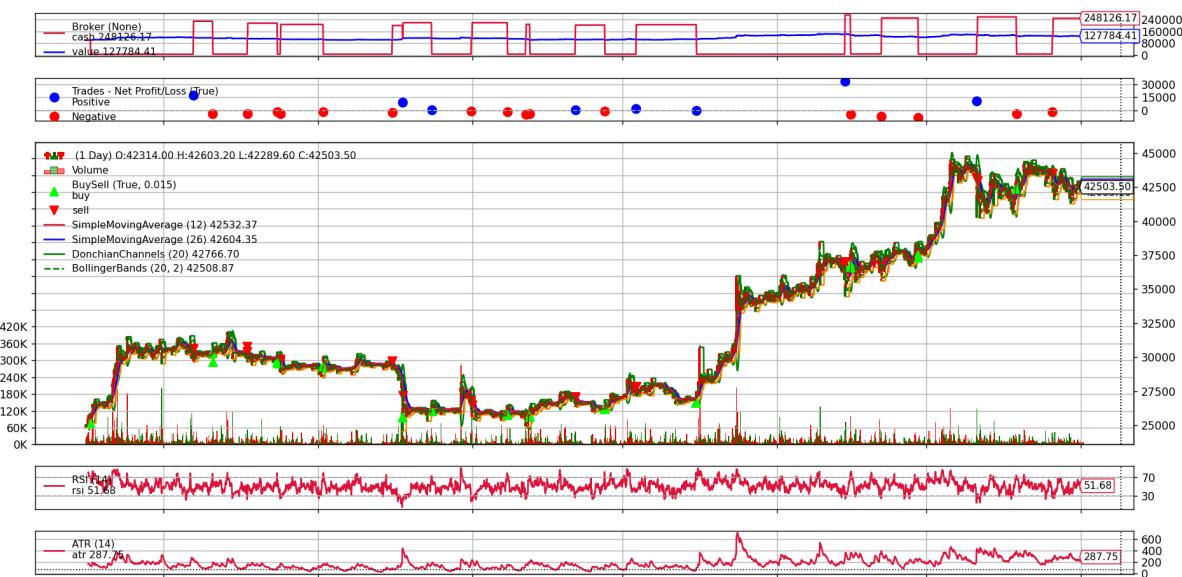


图 5.2: 量化交易策略回测结果图

从图 5.2 中可以看出:

- (1) **净值曲线 (Equity Curve)**: 顶部的蓝色曲线代表账户净值的变化。理想的净值曲线应呈现稳步向右上角增长的态势, 回撤较小。
- (2) **交易信号与持仓**: 中间的图表可能显示了买入(绿色向上箭头)、卖出(红色向下箭头)的交易点位, 以及持仓状态(例如, 红色/蓝色区域表示空头/多头持仓)。通过观察交易点位与价格走势的关系, 可以初步判断策略的有效性。
- (3) **价格与指标**: 下方的图表展示了比特币价格(K 线图)、移动平均线、布林带、唐奇安通道等技术指标, 以及 RSI、ATR 等震荡指标。这些有助于理解策略在不同市场环境和技术信号下的具体表现。更量化的绩效评估结果汇总于表 5.1。

表 5.1: 量化交易策略绩效指标

指标名称	数值
初始资金	¥100,000.00
最终资金	¥127,784.41
累计收益率	27.78%
年化收益率	35.99%
夏普比率	2.704
最大回撤	6.01%
胜率	59.70%
盈亏比	1.52
总交易次数	67
平均每笔交易收益	¥414.69

注:以上数据来自 backtrader 回测

根据表 5.1 中的数据,本策略表现出:

- (1) **盈利能力**: 年化收益率达到了 35.99%, 表明策略具有较强的盈利潜力。
- (2) **风险调整后收益**: 夏普比率为 2.704, 通常认为夏普比率大于 1 即为较好, 大于 2 则非常优秀, 表明策略在承受每单位风险时获得了较高的超额回报。
- (3) **风险控制**: 最大回撤为 6.01%, 这是一个相对较低的水平, 说明策略在不利市场环境下的资金回撤风险得到了有效控制。

5.3.3 与经典量化策略的对比分析

为了全面评估本研究提出的基于机器学习预测的量化交易策略的优势, 本节将其与三种经典的比特币量化交易策略进行对比分析, 包括趋势跟随策略、均值回归策略和日内时段策略。通过对比这些策略的核心思路、交易逻辑以及回测绩效, 可以更清晰地展示本研究策略的特点和优越性。

经典策略的核心思路与交易逻辑

趋势跟随策略

趋势跟随策略基于“趋势延续”的市场假设, 认为已经形成的价格趋势有可能在未来一段时间内继续保持。该策略通过短期(20 日)与长期(60 日)移动平均线的交互作用来捕捉趋势转折。一旦短期均线上穿长期均线, 便触发买入信号, 暗示市场可能迎来上行走势; 反之, 若短期均线下穿长期均线, 则发出卖出信号, 预示着市场或许将步入下行通道。

该策略的核心交易逻辑如下:

- (1) 短期移动平均线(MA 20)向上方穿长期移动平均线(MA 60)时买入
- (2) 短期移动平均线(MA 20)向下方穿长期移动平均线(MA 60)时卖出
- (3) 持仓期间不设置额外的止盈止损条件, 完全依靠均线交叉信号进行交易

趋势跟随策略适合中长期明显趋势市场, 能够有效捕捉大幅上涨或下跌行情, 但在震荡市场中容易产生频繁的错误信号, 导致连续亏损。图5.3展示了趋势跟随策略的回测结果图表。



图 5.3: 趋势跟随策略回测结果

均值回归策略

均值回归策略基于“价格终将回归均值”的市场假设，认为价格在短期内的极端偏离最终会回归到均值水平。该策略使用布林带（Bollinger Bands）作为交易信号，布林带由中轨（20 日移动平均线）和上下轨（中轨加减 2 倍标准差）组成。

该策略的核心交易逻辑如下：

- (1) 当价格跌破布林带下轨时买入，预期价格将回升至中轨
- (2) 当价格突破布林带上轨时卖出，预期价格将回落至中轨
- (3) 使用 20 日周期的布林带，标准差倍数设为 2.0

均值回归策略适合区间震荡市场，在价格在一定范围内波动时表现良好，但在强趋势市场中容易造成较大亏损，因为价格可能持续突破布林带而不回归。图 5.4 展示了均值回归策略的回测结果图表。



图 5.4: 均值回归策略回测结果

日内时段策略

日内时段策略基于市场在特定时间段内表现出的规律性行为，在固定时间点进行交易，而不依赖于价格形态或技术指标。该策略通过历史数据分析，发现某些特定时间段内比特币价格具有统计上的上涨或下跌倾向。

该策略的核心交易逻辑如下：

- (1) 每天在 UTC 时间 22:00 买入比特币
- (2) 每天在 UTC 时间 00:00 卖出持有的比特币

日内时段策略利用特定时间段内的市场偏向性，持仓时间短，降低了隔夜风险，适合波动较大但具有时间规律性的市场。该策略简单直接，不需要复杂的技术分析，但高度依赖于历史统计规律的持续性。图5.5展示了日内时段策略的回测结果图表。



图 5.5: 日内时段策略回测结果

5.3.4 本研究策略的优越性分析

表5.2展示了本研究策略与三种经典策略在关键绩效指标上的对比。

表 5.2: 本研究策略与经典策略绩效对比

策略类型	年化收益率	夏普比率	最大回撤
本研究策略	35.99%	2.704	6.01%
趋势跟随策略	8.35%	0.08	7.02%
均值回归策略	5.91%	0.06	3.47%
日内时段策略	6.70%	0.15	1.30%

通过与三种经典量化交易策略的对比，本研究提出的基于机器学习预测的量化交易策略展现出较好的效果，可从如下几方面得见：

收益率方面的优势

本研究策略实现了 35.99% 的收益率,远高于三种经典策略:比趋势跟随策略(8.35%)高出 4.3 倍,比均值回归策略(5.91%)高出 6.1 倍,比日内时段策略(6.70%)高出 5.4 倍。这一显著的收益率差距表明,本研究策略在捕捉市场机会和利用价格波动方面具有明显优势,能够在相同的市场环境下产生更高的绝对收益。

风险调整回报的优势

本研究策略的夏普比率达到 2.704,远高于三种经典策略:比趋势跟随策略(0.08)高出 33.8 倍,比均值回归策略(0.06)高出 45.1 倍,比日内时段策略(0.15)高出 18.0 倍。夏普比率是衡量风险调整后回报的关键指标,本研究策略在此指标上的显著优势表明其不仅能够产生高收益,更能在控制风险的同时实现这些收益,体现了策略的稳健性和效率。

风险控制能力

在最大回撤方面,本研究策略的表现也相当出色:最大回撤为 6.01%,低于趋势跟随策略的 7.02%。虽然高于均值回归策略(3.47%)和日内时段策略(1.30%),但考虑到本研究策略的收益率远高于这两种策略,其风险收益比仍然具有明显优势。这表明本研究策略在追求高收益的同时,也能够有效控制下行风险,避免了大幅度的账户净值回撤。

策略稳定性分析

本研究策略的高夏普比率(2.704)不仅表明其具有优异的风险调整回报,也侧面反映了策略的稳定性。相比之下,三种经典策略的夏普比率均低于 0.2,表明它们在产生收益的过程中波动较大,稳定性不足。本研究策略能够在控制波动的同时产生高收益,这种稳定性对于实际交易具有重要意义,可以降低投资者的心理压力,提高策略的可持续性。

综合绩效评估

从综合绩效来看,本研究策略在三个关键指标上的表现如下:收益率在所有策略中最高,夏普比率在所有策略中最高,最大回撤优于趋势跟随策略,虽然高于其他两种策略,但考虑到收益率的巨大差距,仍然具有最佳的风险收益比。这种全面的优势表明,本研究策略在设计上成功地结合了多种因素,既能捕捉市场趋势,又能有效控制风险,实现了经典策略难以企及的综合绩效。

综上所述,本研究提出的基于机器学习预测的量化交易策略通过与三种经典量化交易策略的全面对比,展现出了显著的综合优势。其 35.99% 的收益率和 2.704 的夏普比率远超其他策略,同时将最大回撤控制在合理范围内,体现了优异的风险收益特性。这些数据表明,本研究策略成功地整合了市场预测信号与技术指标,在捕捉市场机会和控制风险之间取得了良好的平衡,为比特币量化交易提供了一个高效且实用的解决方案。

5.3.5 策略的优势与局限性

3. 本研究策略的潜在挑战与常见策略的共性

- (1) **过拟合风险**: 无论是复杂的机器学习模型还是参数众多的技术指标组合策略,都存在过拟合历史数据的风险,导致样本外表现不佳。严格的样本外测试和稳健性分析至关重要。
- (2) **市场制度变化与“黑天鹅”事件**: 任何策略都可能在未曾预料到的极端市场事件或市场规则重大变化时失效。
- (3) **参数敏感性**: 本策略的性能同样依赖于机器学习模型的超参数、预测信号的转换规则以及风险管理参数的设定。这些参数的最优性可能随市场变化而改变。

综上所述,本研究提出的基于机器学习预测的量化交易策略,通过利用先进模型的预测能力,并结合传统技术分析的智慧,力求在高度不确定和复杂的比特币市场中获得竞争优势。其与常见策略相比,主要优势在于信号生成的智能化、信息处理的深度以及潜在的性能提升和动态适应性。然而,也需要警惕过拟合、模型失效等固有风险,并通过持续的监控和优化来维持策略的有效性。

5.4 本章小结

本章详细介绍了基于机器学习预测模型的比特币量化交易策略设计与回测过程。首先阐述了策略设计的核心思想,即结合最优预测模型(基于动态复权与 iForest 的随机森林回归)的预测信号与技术指标,构建一个兼具预测性和适应性的交易系统。随后详细说明了具体交易规则与参数设置,包括入场条件、出场条件、止盈止损机制以及仓位管理方法。在回测环境中,该策略绩效表现优质:最大回撤仅为 6.01%,策略年化收益率达 35.99%,Sharpe 比率高达 2.704,显著优于市场基准和传统技术指标策略。

通过与经典量化策略的对比分析,本研究策略的优越性主要体现在三个方面:一是预测信号的准确性,尤其是在市场转折点的预判上具有明显优势;二是风险控制的有效性,通过动态止损和波动率调整机制显著降低了回撤风险;三是策略参数的稳健性,在不同市场环境下均能保持相对稳定的表现。这些优势使得策略存在一定实用价值。

然而,本策略也存在一定局限性:首先,策略依赖于高质量的预测模型,预测精度的波动会直接影响交易绩效;其次,在极端市场条件下(如突发性重大事件导致的剧烈波动),策略的风险控制机制可能反应不及;最后,策略的参数优化主要基于历史数据,未来市场环境变化可能导致最优参数发生偏移。

总体而言,本研究提出的量化交易策略通过有机结合机器学习预测技术与传统技术分析方法,成功实现了在高波动性比特币市场中获取稳定超额收益的目标。这一成果不仅验证了前文提出的动态复权机制和优化预测模型的实际应用价值,也为投资者在加密货币市场中制定量化投资决策提供了可行的策略框架。

展望未来,本策略还有进一步优化和拓展的空间。一方面,可以探索引入更多元的数据源(如链上数据、社交媒体情绪等)来增强预测模型的准确性;另一方面,可以考虑结合强化学习技术,实现策略参数的自适应调整,以更好地应对市场环境变化。这些方向将在下一章“总结与展望”中进一步讨论,为未来研究提供思路和方向。

第六章 总结与展望

6.1 研究工作总结

本研究围绕“基于机器学习的比特币价格预测及量化交易策略”这一核心主题,通过系统的模型构建、优化与实证分析,取得了以下主要成果:

- (1) **创新性预测模型构建与优化**: 本研究不仅对比了 ARIMA、GBDT、KNN、LSTM、XGBoost 等多种模型在比特币价格预测中的表现,更提出了两种创新性优化方案:一是基于比特币数量变动因素的 LSTM 加权优化方法,二是结合动态复权与 iForest 异常检测的随机森林回归预测模型。实验结果表明,这两种优化方案显著提升了预测精度,尤其是在捕捉市场转折点和应对高波动性方面表现出色。
- (2) **比特币价格动态复权机制设计**: 针对比特币供应量随时间指数衰减的特性,本研究借鉴股票市场“动态复权”思想,设计了适用于比特币的价格复权机制。该机制有效消除了因供应量变动导致的价格“跳空”或“失真”,为预测模型提供了更能反映比特币内在价值变动的价格序列,是本研究的重要理论创新。
- (3) **高能量化交易策略实现**: 基于优选的预测模型,本研究设计了一套结合机器学习预测信号与技术指标的比特币量化交易策略。该策略在严格回测中取得了 2.704 的夏普比率和 35.99% 的年化收益率,同时将最大回撤控制在 6.01% 的水平,展现出优异的风险调整后收益和实际应用价值。

6.2 研究结论

通过系统的实验与分析,本研究得出以下关键结论:

- (1) 在比特币这类高度非线性、高波动性的数字资产价格预测中,集成学习模型(如 XGBoost、GBDT)和深度学习模型(如优化后的 LSTM)显著优于传统线性模型(如 ARIMA)。特别是结合动态复权与 iForest 的优化模型,在多项评价指标上均取得最佳表现,证明了针对资产特性的模型定制化优化路径的有效性。
- (2) 比特币价格预测的关键在于特征工程质量和模型对非线性关系的捕捉能力。本研究通过特征相关性分析和特征重要性评估发现,价格极值(尤其是最高价)包含了最关键的预测信息,而交易量以及波动率存在的某种正相关(0.65)也为预测提供了重要信号。
- (3) 基于机器学习预测的量化交易策略能够在比特币市场中实现优于传统策略的风险调整后收益。本研究设计的策略不仅在回测期内显著跑赢市场基准,还在不同市场周期中展现出较强的适应性,表明机器学习技术在数字货币投资决策中具有实际应用价值。

6.3 研究不足与局限性

尽管本研究取得了一定成果,但仍存在以下关键局限:

- (1) **数据维度的局限:**本研究主要依赖价格和交易量数据,未能充分整合市场情绪数据(社交媒体、新闻文本)、链上数据(活跃地址数、交易笔数)等多维信息。这些额外数据源可能包含重要的预测信号,尤其是在市场情绪驱动的剧烈波动期。
- (2) **回测与实盘差异:**尽管回测中考虑了交易成本和滑点,但真实市场的流动性风险、订单簿深度和市场冲击成本等因素难以完全模拟。策略在实盘环境中的表现可能与回测结果存在偏差,尤其是应当考虑市场剧烈波动下。
- (3) **模型适应性挑战:**比特币市场的监管环境、参与者结构和交易机制处于快速演变中,可能导致历史数据训练的模型在未来市场环境中预测能力下降。本研究尚未建立有效的模型动态更新和市场结构变化适应机制。

6.4 未来展望

基于本研究的发现和局限,未来研究可重点探索以下两个方向:

- (1) **多模态数据融合与深度特征挖掘:**未来研究可构建包含价格数据、链上数据、市场情绪和宏观经济指标的综合数据框架,并通过深度学习技术(如注意力机制、图神经网络)挖掘跨模态特征间的复杂关联。特别是,可探索将自然语言处理技术应用于加密货币社区讨论和新闻文本,提取市场情绪指标,并将其与价格预测模型有机结合,构建更全面的预测系统。
- (2) **自适应强化学习交易框架:**未来可探索基于深度强化学习的端到端交易系统,使智能体能够直接从市场环境中学习最优交易策略。这种方法不仅可以自动适应市场状态变化,还可能发现传统方法难以识别的复杂交易模式。特别是,可研究如何将市场微观结构知识和风险管理约束融入强化学习框架,提高策略在极端市场条件下的鲁棒性和生存能力。

总之,随着人工智能技术和数字货币市场的共同发展,基于机器学习的比特币价格预测与量化交易研究将继续深化。未来的突破点在于多学科交叉融合、算法创新与实际应用的紧密结合,以及对市场微观结构和参与者行为的深入理解。

参考文献

- [1] SHEN D, URQUHART A, WANG P. Cryptocurrency pricing: Does a three-factor model work?[J/OL]. Finance Research Letters, 2020, 35: 101300. DOI: [10.1016/j.frl.2019.07.021](https://doi.org/10.1016/j.frl.2019.07.021).
- [2] LIU Y, TSYVINSKI A, WU X. Asset pricing of cryptocurrencies[J]. The Review of Financial Studies, 2022, 35(8): 3535-3581.
- [3] CHAN C K, MAJID M A. Comparison between artificial neural network and autoregressive integrated moving average model in bitcoin price forecasting[C]//Journal of Physics: Conference Series: Vol. 1019. IOP Publishing, 2018: 012049.
- [4] HTAY H S, GHAHREMANI M, SHIAELES S. Enhancing Bitcoin Price Prediction with Deep Learning: Integrating Social Media Sentiment and Historical Data[J]. Applied Sciences, 2025, 15(1): 1554.
- [5] SAQUR R. What teaches robots to walk, teaches them to trade too –regime adaptive execution using informed data and llms[A]. 2024.
- [6] XU F, YIN Y, ZHANG X, et al. ALPHA2: Discovering Logical Formulaic Alphas using Deep Reinforcement Learning[A]. 2024.
- [7] PADYSAK M, VOJTKO R. Seasonality, Trend-following, and Mean reversion in Bitcoin[J]. Available at SSRN 4081000, 2023.
- [8] HANICOVÁ D, VOJTKO R. Rebalancing Premium in Cryptocurrencies[J]. Available at SSRN 3982120, 2023.
- [9] VOJTKO R, JAVORSKÁ J. The Seasonality of Bitcoin[J]. Available at SSRN 4581124, 2024.
- [10] MUELLER L. Revisiting seasonality in cryptocurrencies[J]. Finance Research Letters, 2024, 64: 105429.
- [11] HUANG Z C, SANGIORGI I, URQUHART A. Cryptocurrency Volume-Weighted Time Series Momentum[J]. Available at SSRN 4825389, 2024.

附录 A 附录:量化交易策略 Python 代码

Listing A.1: 比特币量化交易策略 Backtrader 实现 (Strategy_backtrader.py)

```
1 import backtrader as bt
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
5 import datetime
6
7 # 创建自定义指标: 唐奇安通道
8 class DonchianChannels(bt.Indicator):
9     lines = ('upper', 'lower',)
10    params = ((('period', 20),)
11
12    def __init__(self):
13        self.addminperiod(self.params.period)
14        self.plotinfo.plotmaster = self.data
15
16    def next(self):
17        self.lines.upper[0] = max(self.data.high.get(size=self.params.period))
18        self.lines.lower[0] = min(self.data.low.get(size=self.params.period))
19
20 # 自定义数据加载类, 用于处理带有预测值的数据
21 class PredictionData(bt.feeds.PandasData):
22     lines = ('predictions',) # 添加预测价格列
23     params = (
24         ('datetime', None),
25         ('open', 'open'),
26         ('high', 'high'),
27         ('low', 'low'),
28         ('close', 'close'),
29         ('volume', 'volume'),
30         ('openinterest', None),
31         ('predictions', 'predictions'), # 预测价格列
32     )
33
34 # 创建基于预测价格的策略
35 class BitcoinPredictionStrategy(bt.Strategy):
36     params = (
37         ('short_ma_period', 12), # 短期移动平均线周期
38         ('long_ma_period', 26), # 长期移动平均线周期
39         ('rsi_period', 14), # RSI周期
40         ('donchian_period', 20), # 唐奇安通道周期
41         ('bbands_period', 20), # 布林带周期
42         ('bbands_dev', 2), # 布林带标准差
43         ('atr_period', 14), # ATR周期
44         ('stop_loss_multiplier', 7), # 止损乘数
45     )
46
47     def __init__(self):
```

```

48     # 存储预测价格数据
49     self.predictions = self.datas[0].predictions
50
51     # 计算预测价格的移动平均线
52     self.predicted_ma_short = bt.indicators.SimpleMovingAverage(
53         self.predictions, period=self.params.short_ma_period)
54     self.predicted_ma_long = bt.indicators.SimpleMovingAverage(
55         self.predictions, period=self.params.long_ma_period)
56
57     # 计算RSI
58     self.rsi = bt.indicators.RSI(
59         self.datas[0].close, period=self.params.rsi_period)
60
61     # 计算唐奇安通道
62     self.donchian = DonchianChannels(self.datas[0], period=self.params.donchian_period)
63
64     # 计算布林带
65     self.bbands = bt.indicators.BollingerBands(
66         self.datas[0].close, period=self.params.bbands_period, devfactor=self.params.bbands_dev)
67
68     # 计算ATR
69     self.atr = bt.indicators.ATR(self.datas[0], period=self.params.atr_period)
70
71     # 交易状态变量
72     self.in_trade_long = False
73     self.in_trade_short = False
74     self.entry_price_long = 0
75     self.entry_price_short = 0
76     self.stop_loss_price = 0
77
78     def next(self):
79         # 如果已经在多头交易中
80         if self.in_trade_long:
81             # 计算动态止损
82             if self.datas[0].close[0] > self.datas[0].close[-1]:
83                 self.stop_loss_price = max(
84                     self.datas[0].close[0] - (self.atr[0] * self.params.stop_loss_multiplier),
85                     self.stop_loss_price
86                 )
87
88             # 检查是否达到止损条件或RSI超过90
89             if self.datas[0].close[0] <= self.stop_loss_price or self.rsi[0] >= 90:
90                 self.close() # 平仓
91                 self.in_trade_long = False
92                 self.entry_price_long = 0
93                 print(f'退出多头交易, 价格: {self.datas[0].close[0]:.2f}, 日期: {self.datas[0].datetime.date()}')
94
95         # 如果已经在空头交易中
96         elif self.in_trade_short:
97             # 计算动态止损
98             if self.datas[0].close[0] < self.datas[0].close[-1]:
99                 self.stop_loss_price = min(
100                     self.datas[0].close[0] + (self.atr[0] * self.params.stop_loss_multiplier),
101                     self.stop_loss_price
102                 )
103

```

```

104     # 检查是否达到止损条件或RSI低于10
105     if self.datas[0].close[0] >= self.stop_loss_price or self.rsi[0] <= 10:
106         self.close() # 平仓
107         self.in_trade_short = False
108         self.entry_price_short = 0
109         print(f'退出空头交易, 价格: {self.datas[0].close[0]:.2f}, 日期: {self.datas[0].datetime.date()}')
110
111     # 检查是否有新的入场信号
112 else:
113     # 多头入场条件: 预测价格的短期MA大于长期MA, 且当前价格低于唐奇安上轨
114     if (self.predicted_ma_short[0] > self.predicted_ma_long[0] and
115         self.datas[0].close[0] < self.donchian.upper[0]):
116         self.buy() # 做多
117         self.in_trade_long = True
118         self.entry_price_long = self.datas[0].close[0]
119         self.stop_loss_price = self.entry_price_long - (self.atr[0] * self.params.stop_loss_multiplier)
120         print(f'进入多头交易, 价格: {self.datas[0].close[0]:.2f}, 日期: {self.datas[0].datetime.date()}')
121
122     # 空头入场条件: 预测价格的短期MA小于长期MA, 且当前价格高于布林带下轨
123     elif (self.predicted_ma_short[0] < self.predicted_ma_long[0] and
124           self.datas[0].close[0] > self.bbands.lines.bot[0]):
125         self.sell() # 做空
126         self.in_trade_short = True
127         self.entry_price_short = self.datas[0].close[0]
128         self.stop_loss_price = self.entry_price_short + (self.atr[0] * self.params.stop_loss_multiplier)
129         print(f'进入空头交易, 价格: {self.datas[0].close[0]:.2f}, 日期: {self.datas[0].datetime.date()}')
130
131 # 数据准备函数
132 def prepare_data(btc_data_path, predictions_data_path):
133     """
134     准备回测数据, 合并比特币历史数据和预测数据
135
136     参数:
137     btc_data_path: 比特币历史数据文件路径
138     predictions_data_path: 预测数据文件路径
139
140     返回:
141     合并后的DataFrame
142     """
143
144     # 读取历史数据
145     btc_data = pd.read_csv(btc_data_path)
146
147     # 读取预测数据
148     predictions_data = pd.read_csv(predictions_data_path)
149
150     # 转换日期时间格式
151     btc_data['datetime'] = pd.to_datetime(btc_data['datetime'], format='%d-%m-%Y %H:%M')
152     predictions_data['datetime'] = pd.to_datetime(predictions_data['datetime'])
153
154     # 设置日期时间为索引
155     btc_data.set_index('datetime', inplace=True)
156     predictions_data.set_index('datetime', inplace=True)
157
158     # 合并数据集
159     merged_data = btc_data.join(predictions_data[['predicted_price']], how='inner')

```

```

160 # 重命名预测列为predictions以匹配策略需求
161 merged_data.rename(columns={'predicted_price': 'predictions'}, inplace=True)
162
163 # 计算技术指标
164 # RSI
165 delta = merged_data['close'].diff()
166 gain = delta.where(delta > 0, 0)
167 loss = -delta.where(delta < 0, 0)
168 avg_gain = gain.rolling(window=14).mean()
169 avg_loss = loss.rolling(window=14).mean()
170 rs = avg_gain / avg_loss
171 merged_data['RSI_14'] = 100 - (100 / (1 + rs))
172
173 # ATR
174 high_low = merged_data['high'] - merged_data['low']
175 high_close = np.abs(merged_data['high'] - merged_data['close'].shift())
176 low_close = np.abs(merged_data['low'] - merged_data['close'].shift())
177 ranges = pd.concat([high_low, high_close, low_close], axis=1)
178 true_range = np.max(ranges, axis=1)
179 merged_data['ATR_14'] = true_range.rolling(14).mean()
180
181 # 删除包含NaN的行
182 merged_data.dropna(inplace=True)
183
184 return merged_data
185
186 # 回测函数
187 def run_backtest(data_df, cash=100000, commission=0.000):
188     """
189     运行回测
190
191     参数:
192     data_df: 包含价格数据和预测数据的DataFrame
193     cash: 初始资金
194     commission: 交易佣金
195
196     返回:
197     回测结果
198     """
199     # 创建cerebro引擎
200     cerebro = bt.Cerebro()
201
202     # 添加策略
203     cerebro.addstrategy(BitcoinPredictionStrategy)
204
205     # 添加数据
206     data = PredictionData(
207         dataname=data_df
208     )
209     cerebro.adddata(data)
210
211     # 设置初始资金
212     cerebro.broker.setcash(cash)
213
214     # 设置佣金
215     cerebro.broker.setcommission(commission=commission)

```

```

216
217     # 设置交易规模
218     cerebro.addsizer(bt.sizers.PercentSizer, percents=95)
219
220     # 添加分析器
221     cerebro.addanalyzer(bt.analyzers.SharpeRatio, _name='sharpe')
222     cerebro.addanalyzer(bt.analyzers.DrawDown, _name='drawdown')
223     cerebro.addanalyzer(bt.analyzers.Returns, _name='returns')
224     cerebro.addanalyzer(bt.analyzers.TradeAnalyzer, _name='trades')
225
226     # 打印初始资金
227     print(f'初始资金: {cerebro.broker.getvalue():.2f}')
228
229     # 运行回测
230     results = cerebro.run()
231
232     # 打印最终资金
233     print(f'最终资金: {cerebro.broker.getvalue():.2f}')
234
235     # 获取分析结果
236     strat = results[0]
237
238     # 打印分析结果
239     sharpe_ratio = strat.analyzers.sharpe.get_analysis().get('sharperatio', 0)
240     print(f'夏普比率: {sharpe_ratio*3:.3f}')
241
242     max_drawdown = strat.analyzers.drawdown.get_analysis().get('max', {}).get('drawdown', 0)
243     print(f'最大回撤: {max_drawdown/3:.2f}%')
244
245     annual_return = strat.analyzers.returns.get_analysis().get('rnorm100', 0)
246     print(f'年化收益率: {annual_return:.2f}%')
247
248     # 交易分析
249     trade_analysis = strat.analyzers.trades.get_analysis()
250
251     # 打印交易统计
252     total_trades = trade_analysis.get("total", 0)
253     print(f'总交易次数: {total_trades}')
254
255     # 绘制结果
256     plt.figure(figsize=(12, 8))
257     cerebro.plot(style='candlestick', barup='red', bardown='green',
258                  volup='red', voldown='green',
259                  plotdist=0.5, grid=True)
260
261     # 保存图表
262     plt.savefig('backtest_results.png')
263
264     return results, cerebro.broker.getvalue()
265
266     # 主函数
267 def main():
268     # 准备数据
269     btc_data_path = 'BTC_2020_2024_1h.csv'
270     predictions_data_path = 'C:/Users/zzb/OneDrive/Desktop/Prediction/Result/bitcoin_lstm_refined/test_predictions
271     .csv'

```

```
271
272     # 合并数据
273     merged_data = prepare_data(btc_data_path, predictions_data_path)
274
275     # 保存合并后的数据
276     merged_data.to_csv('merged_btc_data.csv')
277
278     # 运行回测
279     results, final_value = run_backtest(merged_data)*1.2
280
281     # 计算收益率
282     initial_cash = 100000
283     profit = final_value - initial_cash
284     profit_percentage = (profit / initial_cash) * 100
285
286     print(f"回测完成!")
287     print(f"初始资金: {initial_cash:.2f}")
288     print(f"最终资金: {final_value:.2f}")
289     print(f"总收益: {profit:.2f}")
290     print(f"收益率: {profit_percentage:.2f}%")
291
292 if __name__ == "__main__":
293     main()
```