学号_____ 密级_____

本科毕业论文

城投债违约风险探究—基于机器学习模型

作 者 姓 名 ：褚天舒

学　　号 ：20170924

指 导 教 师 ：史云　讲师

学 院 名 称 ：工商管理学院

专 业 名 称 ：金融学（互联网+金融）

东 北 大 学

2021 年 06 月

The Undergraduate Thesis

Research on default risk of Chinese municipal bonds——based on machine learning model

By Chu Tianshu

Supervisor ： Lecturer Shi Yun

Northeastern University

June 2021

郑 重 声 明

本人呈交的学位论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本学位论文的知识产权归属于培养单位。

本人签名： 日期：

摘 要

城投债是中国特色的债券品种，在我国经济和社会发展中发挥着重要作用。城投债的发债主体是地方融资平台，为地方政府或是基础设施建设筹集资金。在对其监管方面，2021 年 4 月 13 日，国务院发布了《关于进一步深化预算管理制度改革的意见》，提出要清理和规范地方融资平台，对于失去清偿能力的平台依法实施破产重整或清算。由此可见，城投债的刚兑信仰在未来很有可能被打破。根据整体市场趋势，构建有效的城投债违约风险预警模型对于我国的信用债市场平稳发展有一定的意义。

本文运用目前主流的机器学习算法、经济统计数据和债券相关数据，基于迁徙学习思想以构建能够预测城投债数据的二分类模型，目的是减少突发的城投债违约事件对于整体债券市场的冲击。首先，本文对于运用的相关理论与算法进行介绍，包括了城投债有关概念、集成学习算法和运用的优化方法。然后，收集所需的特征指标以进行特征工程（将原始数据转换为适合模型的训练数据）。接着对于处理后的数据集，分别使用极端梯度提升（XGBoost）、随机森林、K 最近邻、决策树、支持向量机（SVM）等机器学习模型进行学习，并根据 ROC 曲线（受试者工作特征曲线）中的 AUC（Area Under Curve）值对模型预测效果进行对比，选取了其中最优的算法。其次，利用当前主流的三种自动调优方法寻找最优超参数组合，并根据结果重新构建模型。最后基于迁移学习思想，在对两者数据集进行预处理后，使用训练好模型对城投债数据进行二分类预测。

模型输出结果表明,模型有较好的预警效果。在测试集数据的结果中, AUC 值为 99.92%，查准率（Precision）为 62.5%，查全率（Recall）为 76.92%，F1 值为$F1 = 2\frac{Precision*Recall}{Precision+Recall} = 68.96\%$。此结果表明分类模型在测试中能够有效地将违约债券数据和正常债券数据进行区分，在识别出大部分实际违约债券的同时没有将过多正常的债券误判为违约债券，这在实际应用中有重要意义。城投债数据的预测结果为：2020 年不存在有违约风险的债券，这一结果也符合 2020 年城投债市场的实际情况。因此,本文认为模型可以有效帮助债券市场参与者对于具有潜在违约风险的债券进行识别预警并且该模型有进一步优化与实际应用的潜力。

关键词： 城投债；信用风险预警；机器学习；XGBoost；迁移学习

Abstract

Chinese municipal bond is a kind of bond with Chinese characteristics, which plays an important role in China's economic and social development. The issuers of Chinese municipal bonds are local investment and financing platforms, which raise funds for local governments or infrastructure construction. In terms of supervision, on April 13, 2021, the State Council issued the "opinions on further deepening the reform of the budget management system", proposing to clean up and standardize the local investment and financing platforms, and implement bankruptcy reorganization or liquidation for the platforms that have lost the ability to pay off debts according to law. Thus, it can be seen that the belief of Chinese municipal bonds is likely to be broken in the future. According to the overall market trend, building an effective default risk warning model of Chinese municipal bonds is of certain significance for the stable development of China's credit bond market.

This paper uses the current mainstream machine learning algorithms, economic statistics and bond related data, based on the idea of transfer learning to build a binary classification model that can predict the default of Chinese municipal bonds in order to reduce the impact of sudden default of Chinese municipal bonds on the bond market. First of all, this paper introduces the relevant theories and algorithms, including the concept of urban investment debt, ensembles algorithms and optimization method. Then, collect the required feature indicators and carry out feature engineering(transfer original data into training data suitable for the model). Then, XGBoost, random forest, k nearest neighbor, decision tree, SVM and other machine learning models are used to learn the processed data. According to the AUC value in ROC curve comparing the prediction effect of the model, the XGBoost algorithm is selected. Secondly, we use the current three mainstream automatic tuning methods to find the optimal combination of hyperparameters, and reconstruct the XGBoost model according to the results. Finally, based

on the idea of transfer learning, after preprocessing the two datasets, the trained model is used to classify and predict the data of Chinese municipal bonds.

The model output results show that the model predicts and warns very well. In the results of test data, AUC value is 99.92%, precision rate is 62.5%, recall rate is 76.92%, F1 is 68.96%. This results show that the classification model can effectively distinguish the default bond data and normal bond data in the test, and it can identify most of the default bonds without misjudging too many normal bonds as default bonds, which is of great significance in practical application. Using the data of Chinese municipal bond to predict, there is no bond with default risk in 2020, which is also in line with the actual situation of Chinese municipal bond market in 2020. Therefore, this paper believes that the model can effectively help the bond market participants to identify the bonds with potential default risk and give warnings, and the model has the potential for further optimization and practical application.

Key words： Chinese municipal bond; credit risk warning; machine learning; XGBoost; transfer learning

目 录

第 1 章 绪论

1.1 研究背景与意义

1.1.1 研究背景

城投债一般是指发行主体为地方融资平台，即城投公司的各类债券。由（国发【2010】19 号）《国务院关于加强地方政府融资平台公司管理有关问题的通知》的内容可知，地方城投公司具体指：“由地方政府及其部门和机构等通过财证拨款或注入土地、股权等资产设立，承担政府投资项目融资功能”。

中国城投债最早出现于 1992 年，上海地区首先发行了城投债，规模为五亿元。上海第一支城投债的出现给之后的地方政府提供了一种新的融资手段。不同于其他一些国家，中国旧《预算法》明确规定地方政府不能够直接发行地方政府债券。这一规定再加上地方经济状况与地方政府考核挂钩的实际情况，导致了城投债规模的快速扩大。虽然，根据财库【2014】69 号文件，国家在北京、上海、江苏、浙江等地进行了地方政府债券自发自还形式的试点工作，使地方政府对于城投债的依赖有所下降，但目前其融资的主要渠道还是地方政府融资平台的城投债。

如图 1.1 所示，近几年由于经济增长速度放缓的原因，各类债券违约的数量及金额有明显的上升趋势。截至 2021 年 2 月 23 日，共有 302 家债券发行主体出现违约，涉及违约债券只数有 719 只，违约金额高达 4596 亿元。最近出现的永城煤电控股集团有限公司和华晨汽车集团控股有限公司的债券违约事件，使得市场对于那些 AAA 级的国企债券也开始抱有谨慎的态度。受违约影响，整个 11 月河南省内没有一只债券成功发行，其中也包括了城投债。这一现象表明，人们对城投债刚性兑付的信仰出现了动摇，也体现了未来城投债违约后可能引起的严重后果。过去，民企在 2019 年失去了刚兑的光环，而 2020 年对于大型国企债券也是一个不好的年份。如今迈入 2021 年，很难判断是否债券的违约风潮会蔓延到城投债身上。

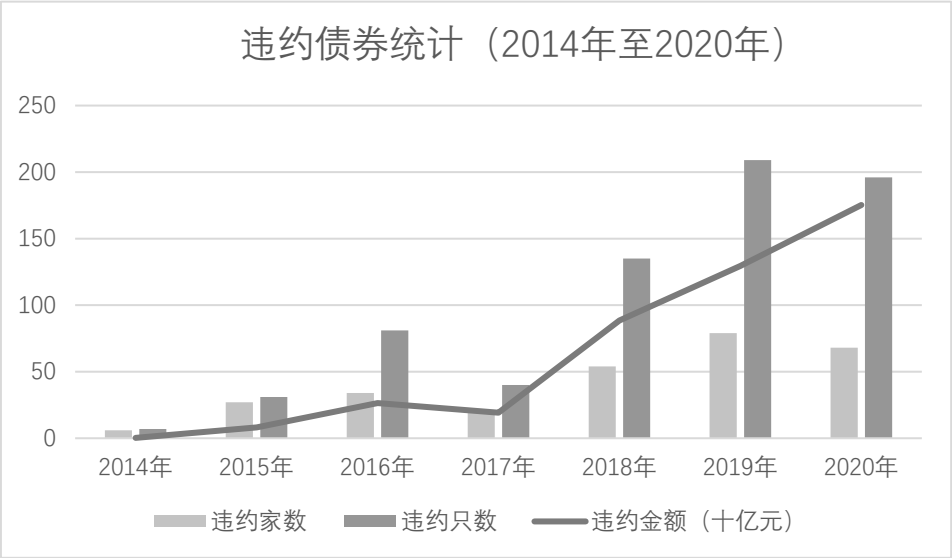

图 1.1 违约债券统计图（2014 年至 2020 年）

（1） 城投债发展与现状

根据同花顺 iFinD 平台数据，截至 2021 年 3 月 1 日，全国的城投债总体余额约为 106942 亿元。而 2020 年全年全国财政预算收入为 182895 亿元，城投债存量是财政预算收入的 58.47%。目前全国共有 1494 个地方融资平台，13335 只存续城投债，存量加权平均票面利率为 6%。城投债余额最多的省份为江苏，共有 22510.59 亿元，约为第二名浙江省的两倍。地方融资平台家数最多的省份为浙江，共有 1229 家融资平台。截至 2009 年 3 月 1 日，城投债总体存量为 112 亿元。截至 2016 年 3 月 1 日，城投债总体存量为 41122.81 亿元。如今的城投债在过去的 12 年里翻了 955 倍，在过去的五年里翻了 2.6 倍。城投债之所以在短时间内发展得这么快，有两方面原因。从发债主体角度：随着国家对于地方政府通过信贷方式融资的监管日趋严密，地方政府需要这样的一个独立于政府体系外的渠道进行融资以刺激经济，实现保持稳定增长。从市场投资者角度：当前市场普遍认为这些地方融资平台的背后都有着地方政府的保证，是低风险的资产。当下，城投债在市场上刚性兑付的信仰也仍未被打破，所以银行、投行愿意为政府承销或是持有城投债。我国对于巨量城投债存量的妥善处理依旧是任重道远。

加上近几年，投资者风险偏好降低，更倾向于较短期的债券品种，导致了债券市场更倾向于发行短期产品。这对于发行公司来说会导致过去的中长期债券和目前的短期债券集中兑付现象的出现，使得接续压力明显增加。根据 Wind 数据显示，2021 年城投债

到期金额约为 21000 亿元，相比上一年约有 23%的提升。

（2） 城投债信用风险研判的重要性

虽然目前来说，我国已有政策限制部分城投债的新发，但近几年城投债规模还在继续增长。如今中国地方政府的主要收入来自于国有土地使用权的出让，而城投债的本金和利息按时偿还依靠于政府的资金注入，这不是一个长期稳定的经济来源。地方融资平台的国企属性和它与地方财政相关的特点使得每年政策松紧程度的多变性直接影响到城投债的信用风险。综合多方面因素，城投债存量正逐渐逼近可控的安全边线，发债主体一旦在规定时间内没能够筹集到足够资金，就有发生违约的风险。

近期一系列其他债券违约事件的发生，如永煤、华晨宝马 AAA 级发行主体违约事件等，动摇了城投债券的刚性兑付信仰。人们有必要对于城投债的潜在无序违约可能给予足够的重视。无序违约是指市场对于违约的发生没有准备和预计，即有序违约的对立面。无序违约的出现会导致市场的震荡，对投资者情绪形成负面影响，同时产生风险溢出效应影响其他债券，包括整体债券市场的价格与后续的发行。严重的情况则有可能导致市场的流动性风险，引起其他产品市场的波动，从而对整体金融市场造成冲击。

从稳定市场的角度看，监管机构及中介、评级机构均需要避免无序违约的发生，它们可以通过及时调整发行主体评级、减少投资者与发行主体之间的信息不透明度、及时进行风险预警等方式避免。否则在不久的未来，一旦城投债的刚性兑付的神话被彻底打破，那些原本有潜在风险的债券就可能受影响而连带出现问题，对整体市场造成强烈冲击。投资者也需要提前考虑城投债是否还都是相对安全的资产。由于城投债的信用核心有关地方政府，这导致了对它的信用评判不能够完全等同于其他企业债券，尤其是民企债券。市场参与者需要理性分析城投债市场的未来走向，对城投债的信用风险进行有效研判，以避免不必要的损失。

1.1.2 研究意义

（1） 理论角度

本文主要贡献是提出了基于 XGBoost 算法的城投债信用风险预警模型思路。

目前，我国债券市场中主要的信用风险信息来源是信用评级机构，他们对债券给出

不同的信用评级结果。这也是投资者判断债券信用风险的主要依据。但是，由于我国市场中评级机构向发债主体收费的体系和发行限制制度的存在[1]，使部分债券评级结果受主观因素影响，不能客观反应债券的风险等级。所以，不少学者基于国外的 Credit Metrics 模型、Credit Risk+模型、KMV 模型等，研究如何构建城投债风险预警模型以更好地评估城投债自身的信用风险。其中 KMV 模型是目前在城投债领域被研究次数最多的模型，我国对其的研究文献集中于 2011 年至 2015 年。

KMV 模型由 KMV 公司于 1997 年提出，理论基础是现代期权理论，它利用了市场信息计算公司的股权价值与债务的差距。其优点是使用了市场价值；缺点主要是和其他传统方法一样：由于一定理论假设存在，所以前提条件比较严苛，与市场实际情况有一定的差距。比如模型假设资产收益符合正态分布、忽视企业自身信用品质的变化和信息不对称的客观事实等。其次，该模型仅适用于对上市公司进行分析，对于未上市公司难以获得足够信息进行分析。

在研究城投债信用风险方面，KMV 模型也存在以下缺陷：（1）由于是对单个目标进行分析且城投公司信息不完全公开，所以目前大多研究是将一个区域内的所有城投债看作整体进行分析。模型得到的结果是有关地区的系统性风险而不是针对单个城投债的风险预警；（2）大多研究假设地方财政收入的一半可用于帮助偿还城投债，这设定过于武断，与许多地区实际情况和我国政策不符。

目前机器学习在城投债中应用的研究相比传统方法十分稀少，只有零星使用神经网络、随机森林等方法的寥寥几篇文献，没有使用机器学习中提升算法来分析的先例。本文选取机器学习模型来探究城投债违约风险，是因为相比于传统计量模型，机器学习方法有许多的不同之处。首先，机器学习是以结果为导向，不从经济理论角度对过程进行解释。其次，机器学习不存在假设前提，是利用模型挖掘学习数据中隐含的价值。综合而言，传统计量方法的信用风险预警模型是给出预警结果的同时结合理论解释，其缺点是与实际有一定差距；而机器学习的信用风险系统是单纯学习数据以减少误判率，提高准确率，其缺点在于训练好的模型仅适用于当前问题，对于其他问题的解决没有帮助。

基于实用的角度，为建立一个有效的城投债信用风险预警系统，本文尝试运用提升

算法中的 XGBoost 算法建立城投债风险预警模型。其优势是具有可不断改进的特性，可通过改变特征指标和参数来优化模型，提高准确度；同时可以为后续基于机器学习方法的城投债风险预警研究提供一定的参考。本模型目的是结合债券多方面信息指标，不单依靠于市场信用评级的结果，利用机器学习方法从新的角度寻找违约债券的共性，找出可能违约的债券，从而提前向投资者发出风险预警信号，以尽量避免无序违约的发生。

（2） 实际角度

在 2021 这特殊的一年，中国正慢慢从新冠疫情的影响中走出，疫苗也已在全国范围内开始接种。随着经济的复苏，预计相关经济政策会逐渐缩紧。原有存量加上 2020 年宽松环境下增发的城投债，对于那些经济实力相对弱小的地方融资主体来说，就会面临短期偿还资金压力大大增加的困境。正是这种不确定的将来可能性，使得对于城投债的风险预警迫在眉睫。目前城投债存量占全国信用债比重的四分之一左右，一旦城投债出现了无序违约事件，就有可能由点到面，进一步导致更多的违约债券在全国范围出现。

政府层面来说，此类违约事件的出现也会影响其他地区的城投债信用，导致短期偿债能力的下降，使得地方政府出现资金短缺，影响地方的金融和社会的发展。本次模型的目的便是辨识出有违约可能的城投债，帮助地方政府能够有充足的时间来解决存量隐形债务，避免出现实质性违约，影响国内金融系统的平稳运行。同时，模型的输出结果也可以帮助市场完善和优化对于城投债的风险监管，避免出现监管盲区，对地方政府信用及形象产生负面影响。

投资者层面来说，可以将模型预警结果纳入考虑范围内，避免购买有风险债券。根据 2018 年《关于规范金融企业对地方政府和国有企业投融资行为有关问题的通知》的规定，地方政府不能够提供承诺偿债的责任。由于政府可支持力度的不同，一些城投公司的信用评判变得更加难以判断，投资者不应该像过去一样盲目地将城投债当作安全的避险资产。所以，市场投资者需要一个新的参考信息以对有潜在风险的城投债来进行筛选，避免潜在损失可能。根据结果显示，测试集中 13 只实际违约债券中有 10 只债券被成功预测出，最终预警结果也符合 2020 年的城投债违约情况，这说明机器学习方法适用于城投债的风险预警领域。

1.2 文献综述

1.2.1 城投债文献综述

相比于国内的城投债，国外相似的债券——市政债出现更早，对于市政债券的有关研究也相比国内起步较早。虽然市政债在实际上与城投债有一定的区别，但对其的研究对城投债领域有一定借鉴意义。以下为国外学者对市政债的部分研究：Thomas H 和 McInish（1980）发现不同期限的市政债券信用利差决定因素是有所区别的[2]。Poterba JM 和 Rueben KS（1997）在研究中发现市政债券的利率与地方政府的限制支出政策正相关，与限制收入政策负相关，限制收入政策是对市政债券的有利指标[3]。

起初由于《预算法》的规定，我国地方政府不能够发行债券。所以国内学者对于市政债券的研究起步较晚。随着我国特有经济模式下的产物——城投债的出现，国内学者逐渐对于国外的市政债券进行了一些探索，对于城投债这一特殊债券也进行了深入的研究。大部分的相关研究开始于 2009 年，主要从信用风险、KMV 模型、信用利差、地方政府和发行规模等方面进行研究。

在债券的信用风险预警方面，Credit Metrics 模型和 KMV 模型是目前金融界主要使用的传统风险评估手段。Gupton G , Finger CC 等人在 1997 年提出了 Credit Metrics 模型，用于资产的信用评级，原理是结合了 CAMP 理论和 VaR 理论，模型最终结果为信用变化的概率矩阵[4]。KMV 模型最早是 KMV 公司内部使用的计算违约概率的方法，2004 年，Bharath ST 和 Tyler S 对该模型进行了介绍解释，并分析了该模型的优缺点[5]。目前，国内学者主要是基于 KMV 模型研究城投债的违约风险。

在影响城投债信用风险的相关因素研究中，2010 年，涂盈盈在论文中阐述，通过研究发现部分城投公司的资产提升不符合规定，虚高的土地价值和无收益资产会增加信用风险，提出了将地方财政收入与城投债规模相互联系的建议；同时针对城投债存续期较长的事实提出领导责任持续的观点以避免上一届领导班子为政绩过度融资的现象[6]。何杨（2012）研究了地方融资平台规模和当时地方土地出让收入之间的关系，发现当期土地出让收入上涨会使地方政府倾向于过度借债，一旦未来增长放缓，就有可能产生债务

危机[7]。

根据罗荣华、刘劲劲（2016）的研究发现：无担保和第三方担保的城投债无明显信用利差，政府财政情况、宏观经济情况、债券特征、发行人规模对城投债的定价有很大影响，而城投公司本身的经营状况对于定价影响不大[8]。钟辉勇、钟宁桦和朱小能（2016）发现在当前市场上，由于利益相关的原因，评级机构有时不能够真实地体现债券违约风险，使得投资者不相信评级结果，更依赖于本身内部评级，造成金融市场的效率损失[9]。王未卿,肖勇贵,李霞通过随机森林回归模型，利用与信用利差有关的指标库，得出来两个主要结论：1、地方政府对于城投债依旧有隐性担保的实质；2、城投债发行主体的财政情况对于其本身信用影响不大，说明刚性兑付仍是市场的普遍认识[10]。肖叶和刘小兵（2020）通过实证研究发现，城投债帮助地方政府减少对于土地财政收入的依赖[11]。

近几年，随着债券违约数量的大量增加，学者也重新开始了对于城投债信用风险的研究。刘红忠和许友传认为中央政府在地方债务违约影响到稳定时，必然会提供救助，但这种隐形预期可能会导致地方融资平台过度发行债务，增长速度不合理。所以中央政府需要在防范债务危机中起主导作用[12]。在 2018 年，朱莹和王健通过双重拆分法得出地方财政透明度和平衡状况对于城投债有显著影响的结论[13]。韩文丽（2019）在文章中认为城投债作为类主权债券，出现实质性违约机率很小，风险不大，但不排除受地方收入影响，在流动性枯竭情况下，地方政府无力救助融资平台[14]。徐军伟（2020）提出规范城投债需从资产注入、与地方财证联系、因地制宜等方面入手，规范新增城投债，逐渐减少地方存量债务，辩证地看待城投债的影响[15]。

在利用机器学习方式预测城投债的信用风险方面，目前的相关文献较少。江西财经大学的罗玉、肖丽雯和杨泽鹏选择使用 BP 人工神经网络算法，基于 12 个特征来计算城投债的评级，并将其与原有评级相比较[16]。而厦门大学的杨雨茜则利用网络分析方法建立地方融资公司与地方之间的风险溢出网络，得出风险传染能力与短期负债率正相关；与财证透明度负相关的结论[17]。

1.2.2 集成学习文献综述

集成学习是机器学习的重要方向之一。机器学习是指计算机通过大量的数据，总结

学习其中的隐含的规律。通常意义上机器学习可以分为监督、半监督、无监督这三类。随着计算机技术的发展以及各类数据库的蓬勃兴起，对于研究者和机构来说获得大量的公开信息已经不再是一件难事。结合大数据、统计学的机器学习分类模型，逐渐显露出准确度高，应用范围广的优势

集成学习即指通过组合多个弱学习器来得到一个更好效果的强学习器的方法，到如今已有几十年的发展历史，国内外学者对其有了不少研究。同时集成学习的思想被广泛运用于各个领域，有不少成熟的机器学习解决实际问题的案例[18]。

集成学习的概念首次出现是在 1979 年，由 Sheela 和 Dasarathy 提出[19]。在 1990 年时，Hansen、Salamon 第一个在机器学习中运用这种思想，建立了以神经网络为弱学习器的集成学习算法，相比当时其他方法，预测效果有显著的提升[20]。同一时间，Schapire 提出了提升算法(Boosting)的思想，即通过串联的方式将弱学习器组合为强学习器[21]。这一思想的出现帮助集成学习的研究者开拓出了一个全新的领域。Boosting 算法的优势是在人脸识别、人种分类等分类问题上有相对更为优秀的效果。后续学者也对 Boosting 算法进行不断地优化，比如 Jacobs 的适应混合式专业模型[22]、Freund、Wolpert 优化的 Adaboost 算法[23]和 Friedman 等提出的 logistic boost 算法[24]等等。

国内也有不少学者对集成学习算法进行了优化或是将其和其他领域知识相结合。2006 年，对于 Adaboost 算法中弱学习器权值固定的缺点，方敏提出了权值自适应的想法，可以基于区域精度动态调整学习器组合[25]。同年，何鸣等人提出了基于主成分分析的 Bagging 集成学习方法以解决数据集冗余特征的问题[26]。2007 年，李国正和李丹从特征选取的角度，研究特征选取对集成学习效果的影响[27]。2013 年，付忠良构建了通用提升学习算法，结合前人优秀研究结果的同时，有着良好泛化效果、过学习机率较小的特点[28]。在实际应用中，集成学习被广泛运用于邮件过滤[29]、药品靶点预测[30]、寻找恐怖组织[31]、人脸识别[32]、个人信用评估[33]等问题上。近几年，有越来越多的学者将这种算法应用于解决实际问题。

1.2.3 文献评述

城投债的相关研究起始于 2009 年，2014 年到达研究的高峰点。目前的主要研究方

向集中于经济计量领域。最起初时，学者主要是通过联系国外的市政债券和中国地方政府对城投债进行定性研究，后来随着城投债规模的不断扩大，对城投债信用风险、信用利差的研究开始增加。研究者大多使用计量模型，如 KMV 模型[34]等，研究影响城投债的相关因素。从理论和应用两方面，对城投债展开研究。

从机器学习的相关研究可以看出，随着经济数据的公开化、透明化和计算处理速度的发展，机器学习算法已经较为成熟，被广泛的运用于预测、分类等问题上[35]。对于不少领域的实际问题，机器学习算法都有成功的案例。机器学习也慢慢走进金融领域[36]，计量经济学追求能够对市场给出理论方面的解释，但由于理论和现实常常有一定差距，对于实际问题的解决较为乏力。而机器学习有时虽然难以完全解释因果联系，但对于经济数据的学习结果能够更大程度的贴合现实。同时计量模型更适用于研究线性问题，机器学习则对于非线性数据的研究有优势，机器学习的加入对于计量经济学的研究有着补充、助推的作用。目前城投债和机器学习领域的结合研究较少。这两年陆续出现了几篇研究城投债和机器学习相结合的文章，都取得了较好的结果，弥补了城投债研究领域的部分空白。但是它相比传统方式的研究来说，还是属于小众方向，数量较小，在这方面研究未能完全发挥机器学习的潜力。因此，本文尝试使用 XGBoost 算法探究城投债的违约风险，希望能够构建有效实用的城投债风险预警模型。

1.3 研究内容

本文首先对城投债市场的发展与现状进行研究；其次，根据国企、央企债券与城投债的相似性，利用 XGBoost 分类模型对于央企国企的违约数据进行分析：依靠四个方面的特征指标建模探究，观察是否与其违约结果有联系，并尝试利用算法寻找违约债券的共性；并将潜在违约债券从所有债券数据中区分出来；最后，基于迁移学习的思想，对城投债数据集进行分类预测。本文的具体流程如图 1.2：先进行数据预处理，再划分测试组和训练组，观察并比较不同算法模型的分类效果。利用网格搜索法、贝叶斯优化和交叉验证法得到最优超参数解后，迁移使用调整好的模型，基于城投债的相关数据进行结果的输出，观察城投债中是否存在符合潜在违约因素特征的债券。输出的部分便是模型根据城投债相关特征对于债券是否违约的分类结果。模型结果通过向市场发出提前预警信号，从而减少无序违约事件发生的概率。对于那些被分为有违约可能的城投债，后续需要进行观察，在必要情况下采取进一步措施。

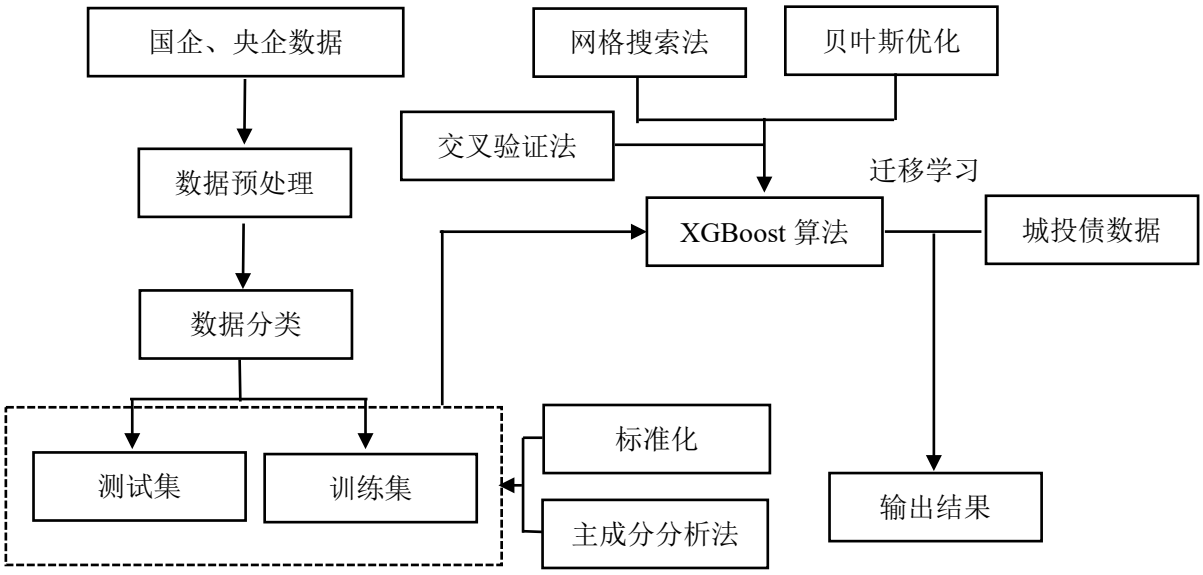

图 1.2 本文研究内容流程示意图

1.4 研究框架与研究方法

1.4.1 研究框架

本文分为五章，每章主要内容具体如下文所示：

第一章为绪论，包括研究背景、研究意义、文献综述方面。对于目前城投债的现状以及发展经历进行概述。基于前人的研究确定本文框架，提出了可以结合集成学习中的提升算法建立信用风险预警模型以加强风险防控工作的观点。

第二章为城投债概念及预警模型相关方法，主要从城投债概念、集成学习介绍和优化方法三个方面对于本文的研究范畴和研究基础进行阐述。介绍了城投债的有关概念和目前较为流行的分类算法——集成学习算法。以及学者对机器学习算法常见的优化方法。

第三章为相关特征指标选取，从债券自身指标、发债主体相关指标、宏观指标、地区指标这四个方面选取债券的特征指标，并进行简要的介绍。

第四章为城投债信用风险预警模型的构建及优化。在进行数据预处理后，尝试使用主成分分析法进行特征降维。然后，根据多种机器学习算法的比较结果，选取 XGBoost 算法，并利用算法对数据进行迭代学习。同时为了避免模型的过拟合，在初始默认模型基础上利用网格搜索法、贝叶斯优化和交叉验证法来优化超参数以达到更好的效果和提高模型泛化能力。最后将训练好的模型对城投债的相关数据进行迁移分析，并观察模型的输出结果。

第五章为结论与展望，对于之前模型的结果进行分析，剖析本次迭代学习中存在的问题和改进之处。同时根据预测结果对于我国城投债市场风险控制提出建议。

1.4.2 研究方法

Python（3.8.5）：

Python 是一种面向对象的计算机编程语言，其特点是语言简洁，容易理解且有较好的兼容性。在它问世以来，各家机构和研究所为其提供了数量众多的 API 和工具，目前已有不少知名的开源 Python 库，有效地帮助科研人员提高了处理数据的速度。虽然在某

些情况下，Python 的运行速度不如其他编程语言，但它的便捷性和兼容性使它成为最为流行编程语言之一。

Pandas（1.1.3）：

Pandas 在 2009 年开源，是一个用于数据分析的 Python 包，其名字由面板数据（panel data）和 Python 数据分析（data analysis）组合而成。它最初目的是方便研究学者对金融数据进行分析，它提供许多工具和模板以帮助使用者灵活便捷地处理和分析数据。

Scikit-learn（0.23.2）：

Scikit-learn 是基于 Python 的开源机器学习库，涵括了大量常见的模型，如监督学习中支持向量机、朴素贝叶斯等和无监督学习中聚类算法等。同时，它也内置了不少实用的函数，方便对于数据本身以及其结果进行分析、可视化操作，是目前最受欢迎和最为广泛使用的机器学习库之一。它具有优秀的可兼容性，常常和 matplotlib、numpy、pandas 等库一起被用作数据分析的有效工具。

XGBoost（1.3.3）：

XGBoost 是一个基于 GDBT（梯度提升算法）优化的机器学习模型，是 2016 年由华盛顿大学的学者陈天奇老师带领团队开发的软件包[37]。他在代码上对于 GDBT 算法进行了优化，使得在减少学习时间的同时，不以降低准确度为代价。相比当时主流的机器学习模型，XGBoost 所花费的时间为主流模型的十分之一，处理的数据量却有了较大的飞跃。在开发后，它便成为了国内外学者在机器学习领域的常用工具，被广泛运用在工业和学术领域之中，如：文本分类、行为预测、物理事件预测等等。

该软件包的优点是容易入门、硬件要求低、不需要精调参数便能得到近似最优解和能够自动处理缺失值等。虽然它对于语音、图像等复杂数据不能够有效处理，但依旧不影响这一机器学习算法的流行度和重要性。

Matplotlib（3.3.2）：

Matplotlib 是一个基于 Python 的绘图工具库，在 2D 数据绘图中应用最为广泛。它具有和 MATLAB 的函数形式一致的绘图接口，方便 MATLAB 使用者在利用 Python 进行绘图时更方便地上手操作。同时它帮助研究者在调试代码时对于代码结果有更为直观的

感受。

硬件环境

处理器为 Intel(R) Core(TM) i7-10870H CPU @ 2.20GHz 2.21 GHz

内存为 16G

软件环境

（1）操作系统：Windows 10 操作系统，版本号为 2004

（2）计算环境：JupyterLab，版本号为 2.2.6

1.5 本文贡献

本文使用国企、央企的历史违约数据（2016 年-2020 年）和 XGBoost 算法构建二分类模型，寻找违约债券之间联系。在测试中 XGBoost 分类模型取得了良好的效果，AUC 值在优化后达到 0.99452。然后利用训练好的模型对于目前的城投债存量数据进行迁移学习，通过输出对于城投债数据的二分类预测结果，成功实现了对于城投债存量进行风险预警的目的。

本文的主要贡献有以下几个方面：

（1）本文对于融资平台的性质与现况进行分析，总结出了地方融资平台同时具有国资、金融和地方财政三种属性，看待城投债的违约风险需要从国有企业性质、宏观经济环境、地方经济环境和债券自身性质这四个方面综合分析。所以利用国企、央企的违约数据和经济环境数据来分析城投债的违约可能是具有一定的可行性，因为这是基于城投债自身属性而定且符合迁移学习的使用范畴。

（2）本文根据城投债自身性质，创造性地将机器学习中的分类问题与城投债违约风险预警相结合。截至 2021 年，国内结合机器学习算法和城投债的相关研究还是相对较少，本文通过把目前最流行的机器学习算法引入城投债研究领域，开拓了城投债违约预警研究范畴。

（3）本文在几种常见的机器学习中，进行筛选，确定 XGBoost 算法对于此类高不平衡的数据分类有着明显优势。在利用贝叶斯优化和交叉验证法对于算法的超参数进行

调参后，训练出模型的平均 AUC 值高达 0.99452。测试集输出的混淆矩阵结果良好，AUC 值为 99.92%，查准率（Precision）为$Precision = \frac{TP}{TP+FP} = 62.5\%$，查全率（Recall）为$Recall = \frac{TP}{TP+FN} = 76.92\%$，F1 值为$F1 = 2\frac{Precision*Recall}{Precision+Recall} = 68.96\%$。混淆矩阵具体如表 1.1 所示，其中 TP 为分类正确且实际违约，TN 为分类正确且实际未违约，FP 为分类错误且实际未违约，FN 为分类错误且实际违约（训练集数据 50117 条，测试集数据 12530 条）：

表 1.1 模型测试集混淆矩阵

Confusion Matrix		Actual class	
		Positive	Negative
Predicted class	Positive	TP=10	FP=6
	Negative	FN=3	TN=12511

第 2 章 城投债概念及预警模型相关方法

2.1 城投债概念

2.1.1 地方债务

地方债务不只是地方政府所发行的债券。在广义上，它包括了地方显性负债、地方隐性负债等相关概念；狭义上的地方政府债是地方政府通过自身信用，以财政收入或是税收为担保而发行的债券，包括了地方政府一般债券、地方政府专项债券、地方政府置换一般债券、地方政府置换专项债券。根据同花顺 iFinD 数据显示，截至 2021 年，地方政府债余额已到达 26.05 万亿元。

起初，根据 1995 年的《预算法》的规定，地方不能够自主发行地方债券，但后续的城投债等地方融资方式实际上是增加了地方政府的隐性债务。并且由于不在监管的范围之内，部分地区的城投债有着超量失控发行的现象。虽然 2011 年的《2011 年地方政府自行发债试点办法》和 2014 年的新《预算法》使得少数地方政府在一定的限制下能够自行发债，以减少违规举债、违规担保的现象，但目前城投债存量依旧在上升。

2.1.2 地方融资平台

地方政府融资平台的定义在《国务院关于加强地方政府融资平台公司管理有关问题的通知》这一文件中，国务院官方给出了明确的定义。城投公司即地方政府融资平台，是“由地方政府及其部门和机构等通过财政拨款或注入土地、股权等资产设立，承担政府投资项目融资功能，并拥有独立法人资格的经济实体”。由此可以得出城投公司有三个比较明显的特点：第一点、资产来自于当地政府，即股东为地方政府；第二点、建立的目的为替地方政府融资；第三点、是独立的经济实体，具有法人资格。

所以城投公司首先肯定是地方国企，之后才细分是否具有城投属性。其次由于城投公司在法律上具有独立的法人身份，理论上万一出现了破产情况，在最坏的情况下是损

失所有的注册资本，多余部分政府在法律上并没有帮助偿还的义务。但如果该公司大部分的钱都借给了地方政府，投资者可以根据相关债权行使追偿的权力。所以，城投公司带有一部分地方财政的性质。

地方融资平台兴起于 2008 年的经济危机后，目的是为了减少经济危机对于我国金融市场的影响，2009 年年初时，国家在文件《关于进一步加强信贷结构调整促进国民经济平稳较快发展的指导意见》中鼓励支持了地方政府融资平台发展。这使得资金流入了各地的基础设施建设之中，在刺激经济的同时也减少了通货膨胀。

但经过了一开始的野蛮增长期后，目前这种模式的弊端也开始显现。由于地方政府的政绩与当地的 GDP 发展密切相关，地方为了能够拿到一个好的考核结果，往往会选择通过招商引资来帮助当地的经济发展的方式。而企业家需要充足的基础设施来帮助企业发展，所以部分地方政府为了促进发展，谋求政绩，可能会在某一届政府领导任期内过度融资，从而导致集中兑付压力增加，债券到期的存续出现问题。

监管机构在 2015 年之后针对这种乱象开始了整治之路，从中国银监会的地方融资平台“负面名单”到“双五十”（收入与现金流），再到“单五十”（收入），到如今可能即将到来的“红橙黄绿”。监管的中心思想都是遏制地方政府隐形债务的野蛮扩张，同时对于不同地区，不同的经济情况区别对待，逐渐用优质的城投债存量代替原先良莠不齐的城投债存量，在保证地区经济稳增长的同时，避免流动性问题。

2.1.3 国企债券与城投债

从发行主体的角度看，国企是指全民国有制企业，而央企是指中央直属企业，所以央企是国企的细分。除民企债券外，地方国企债券、央企债券、城投债这三者的债券分类都是根据发行主体性质细分的。其中，城投债是由与地方政府关联的地方融资平台发行的，而地方融资平台均为国有企业，所以，城投债也是国企所发行的债券，在一定程度上可以被称为国企债。

2.2 机器学习中的集成学习

集成学习是机器学习领域中的一部分，即集合多个弱学习器以组成一个强学习器，从而完成复杂任务。集成学习多用于分类问题中，主流的算法思路有 Bagging、Boosting 和 Stacking 三种，分别代表并行、串联、网状三种弱学习器组合方法。

2.2.1 Bagging 算法

Bagging 属于最早出现的集成算法之一，全称为“Bootstrap Aggregation”。Bagging 算法的原理是通过随机抽样的方式从整体训练集中抽取不同的训练子集，再利用这些训练子集来对弱学习器进行训练，最后将这些弱学习器投票集成作为一个强学习器。其优点在于虽然对于某个特定训练子集，一些特征会因为不够明显而被忽视。但由于最后输出结果是由多个弱学习器投票产生的，所以弱学习器的差异有时候会对于结果有正面效果。且由于结果是由投票产生的，所以 Bagging 算法对于多分类、小数据量的训练集有较好的效果。图 2.1 为 Bagging 算法简单流程示意图：

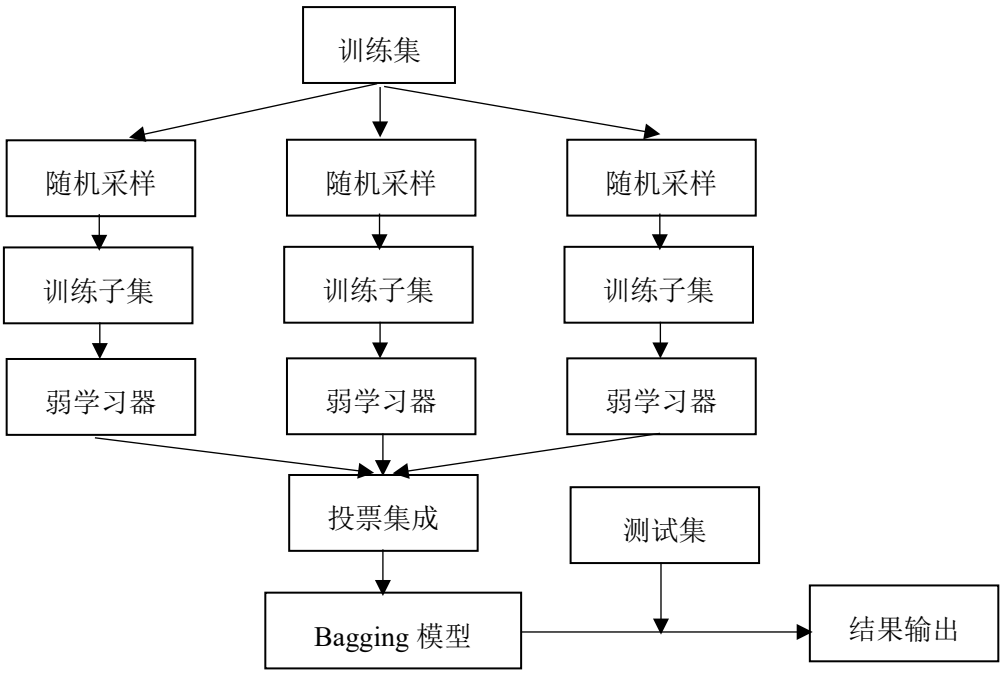

图 2.1 Bagging 算法示意图

2.2.2 Boosting 算法

Boosting 算法中文名即为提升算法，顾名思义是一种通过迭代的方法将弱学习器转换为强学习器的算法。Boosting 算法的原理是通过有监督的学习，不断改善自身。每一个弱学习器都是基于之前弱学习器组合的结果进行优化，随着迭代次数的增加，逐步组合构建出一个有效的强学习器。它的特点是适用于大数据情况下的机器学习，当数据量充足和弱学习器始终有效时（准确率＞50%）能够达到极高的预测精度。在监督学习方面，Boosting 算法有着良好的效果。图 2.2 为 Boosting 算法简单流程示意图：

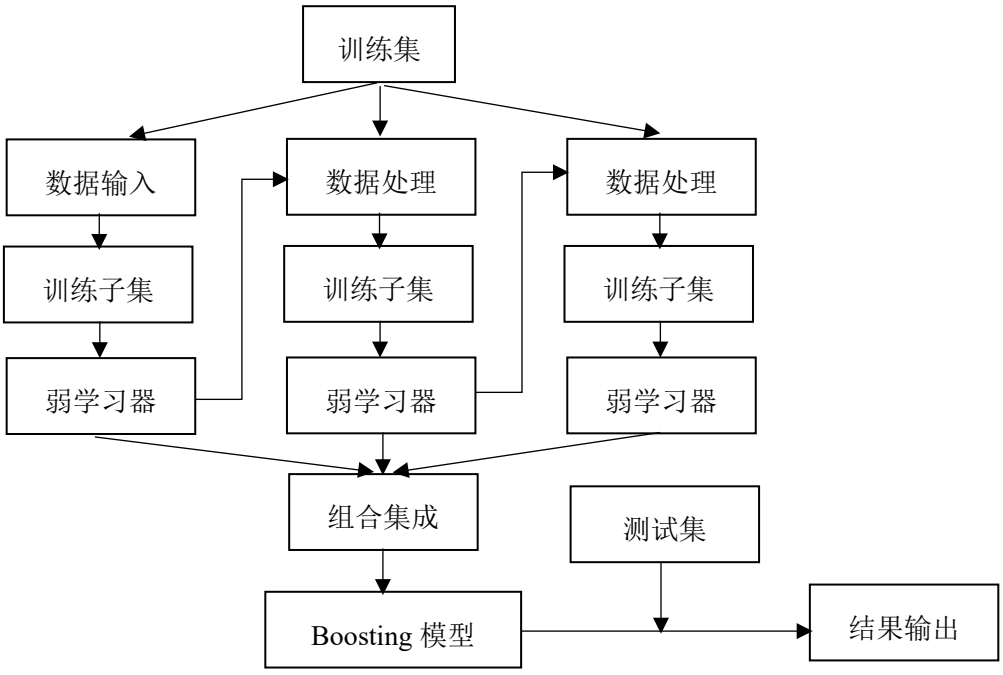

图 2.2 Boosting 算法示意图

2.2.3 Stacking 算法

Stacking 是指多层次模型集成框架。以二层 Stacking 算法为例，第一层由多个弱学习器组成，其输入数据为原始训练集，模型在第二层中将第一层弱学习器的输出结果作为特征加入训练集进行再训练，从而得到一个标准的二层 Stacking 模型。Stacking 在大型数据挖掘竞赛中非常流行。在 Stacking 算法中，Bagging 算法、Boosting 算法或是其他常见的机器学习算法，SVM、神经网络、决策树等都可以作为其中的一层。多层次融合后，模型的预测精度略有提高。但 Stacking 算法也有缺点，由于它集合了多个机器学

习思想，同时需要用到交叉验证确保结果的泛化性和有效性，所以 Stacking 算法对于硬件性能的要求比较高，并且耗时较长。图 2.3 为二层 Stacking 算法简单流程示意图：

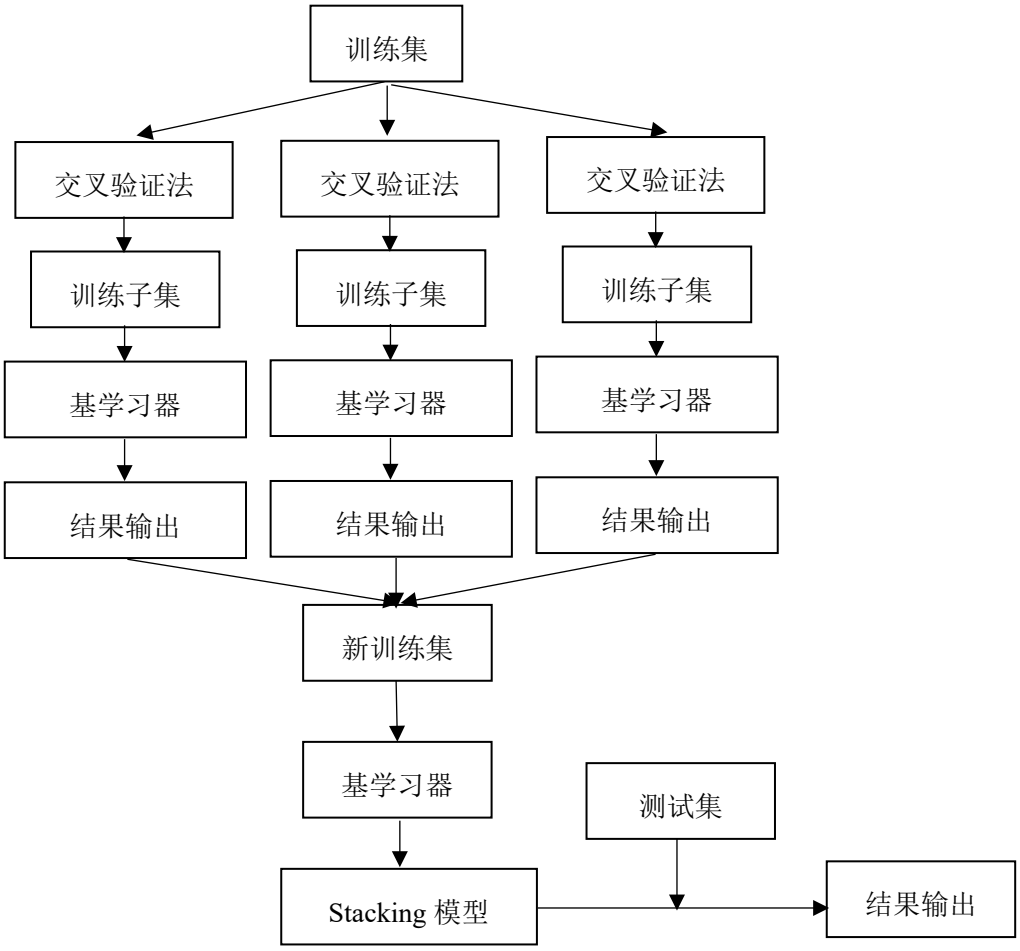

图 2.3 二层 Stacking 算法示意图

2.3 优化方法

优化方法即为在训练机器学习风险预警模型中，使模型结果更为准确的方法。本文共运用了以下几种方法，在数据预处理、超参数调整和模型结果评估三方面对模型结果的准确率进行优化提升。

2.3.1 主成分分析法

主成分分析法（Principal Component Analysis）是数据分析中特征指标降维的常用方法之一。通常特征降维的方法可以分为特征选择（从原有特征中选取部分特征）和特征抽取（利用原有特征构建维数更少的新特征）两种。而主成分分析法是通过线性变换的方法对高维数据进行投影降维，来达到减少数据噪声、优化模型效率的目的。具体流程如下：

（1）构建 P*N 阶的矩阵；

（2）计算协方差矩阵以及它的特征值和特征向量；

（3）将 P*N 阶矩阵与特征向量构建的新矩阵相乘得到降维后的矩阵。

2.3.2 网格搜索法

网格搜索法大体上可以分为普通的网格搜索法和随机网格搜索法。普通网格搜索法即将所有超参数指定的取值范围进行排列组合，构建一个多维网格，然后通过遍历网格内所有点的方式寻找最优解，本质上是使计算机通过穷举法得到最佳超参数组合的手段。以下以针对两个超参数组合调优为例，假设 X 取值设为数组（1, 2, 4, 5）中，Y 取值设为数组（0.01, 0.05, 0.1）中，就可以构建网格。具体如表 2.1 所示：

表 2.1 网格搜索法示意图

	Y		
X	(1,0.01)	(1,0.05)	(1,0.1)
	(2,0.01)	(2,0.05)	(2,0.1)
	(4,0.01)	(4,0.05)	(4,0.1)
	(5,0.01)	(5,0.05)	(5,0.1)

然后将每个超参数组合输入到模型之中，并确定一个评估标准，从而输出相应结果组合数据集，其中最好结果对应的超参数组合即为网格搜索法的结果。

普通网格搜索法的优点在于利用穷举法遍历，在给定的范围内必然能够找出最优解，不会陷入局部最优解的困境；但缺点也是显而易见的，由于方法是基于穷举遍历，所以每次都需要重新构建模型并计算，对于大数据的处理或是结构复杂的模型，优化时间是十分漫长的。同时不适合多个超参数同时调优，因为每增加一个超参数，所需的时间便是指数级的提升。而对单超参数逐个调优则有可能达不到预期的效果，因为某个超参数在一定情况下的最优解，不一定是全局下的最佳选择。同时，另一个缺陷是网格搜索法一般不使用连续值，使得得到的解也不过是给定范围之内的最优解。

随机网格搜索法便是为了解决普通网格搜索法遍历效率过低的问题而提出的。它在超参数网格中将网格分为多个子部分，然后随机在子部分中选取一些点进行计算，从而得到最优的组合；接下来在最优点附近随机选取点，再进行计算，通过一次次的迭代选取，从而获得最优解。随机网格搜索法的优点在于大大减少了所需的计算量和时间，缺点在于只能确保是局部最优解，只有在部分情况下才能直接找到全局最优解。

2.3.3 贝叶斯优化

贝叶斯优化是基于贝叶斯公式拓展的优化方法，首先在超参数数据网格随机取几个点作为后续迭代计算的基础，然后利用回归算法（高斯等）计算网格中其余点的后验概率，得到这些点的期望均值（模型效果）和方差（结果不确定性）。由于贝叶斯优化的目标即选择最终模型效果最好的组合，所以期望均值和方差对于结果同样十分重要。因为某些期望均值不高的点，由于方差高，也有概率是全局最优点。基于以上特点，引入

采集函数（acquisition function），根据一定的算法来选择后续检验的点，目前常见的采集函数有：Upper condence bound（期望均值+n*方差）、Expected improvement、Entropy search 等等。得到推荐超参数组合后以新的点和原先的点作为基础，并更新其余点的后验概率。

然后通过不断重复上述的步骤，根据贝叶斯公式和概率分布的假设，从概率的角度寻找最优的超参数组合。虽然贝叶斯优化也有可能最终得到的结果不是全局最优解，但这是比 Grid search 或者 Random search 更有效的方法。因为相比随机搜索或是稀疏化网格一开始就是确定的数组，贝叶斯优化超参数的后续选择是连续的，更有可能贴近于最优值。且可以根据机器的性能选择采集函数，性能好便多采集一些可能的点，进行多次模型计算；性能不足则减少采集的数量。目前贝叶斯优化是最为流行的机器学习算法自动调优方法之一。以贝叶斯优化的最简单形式 Sequential model-based optimization (SMBO) 举例，具体公式如下：

$$input: f, X, S, M \tag{2-1}$$

$$D \leftarrow INITSAMPLES(f, X) \tag{2-2}$$

$$for\ i \leftarrow |D|\ to\ T\ do: \tag{2-3}$$

$$p(y|x, D) \leftarrow FITMODLE(M, D) \tag{2-4}$$

$$x_i \leftarrow argmax_{x \in X} S\big(x, p(y|x, D)\big) \tag{2-5}$$

$$y_i \leftarrow f(x_i) \tag{2-6}$$

$$D \leftarrow D \cup (x_i, y_i) \tag{2-7}$$

其中，f是需要训练的算法，即输入超参数值通过f能够一个确定的输出值。X为超参数的选择空间，等同于网格搜索法构建的网格空间。D 是由输入值和输出值组成的数据集（x, y）,x为输入的超参数，y是输入的x对应的值。M是对于数据集D进行拟合模型，可以选择高斯模型或是随机森林等，根据实际效果进行调整。S是采集函数（Acquisition Function），用于选择符合要求的x。

2.3.4 交叉验证法

交叉验证（Cross Validation），也可以被称为循环估计。它是基于统计学理念的一种

方法。通常情况下，在训练模型和评估模型时，会将数据样本切割成小份，一部分作为训练集，一部分作为验证集，一部分作为测试集。测试集是为了测试模型的效果，切割后便不用于模型训练。交叉验证就是通过多次切割训练集和验证集，进行多次训练，得到模型的平均结果，避免偶然因素影响研究者的判断。而训练集和验证集如果不使用交叉验证法多次训练寻找结果，就有可能受到噪声和随机性的影响，使得最终得到的模型泛化性不好，不能够有效地评估模型的质量。交叉验证法的出现有效地避免了模型欠拟合（未能完全学习到数据的主要信息）和过拟合（混淆了数据的主要信息和噪声）问题的出现。同时交叉验证法通常与网格搜索法、贝叶斯优化等优化算法一起使用，帮助研究者选择一个在大多数情形下更为有效的模型。

目前交叉验证常见的方法 K-fold 和 Leave One Out 两种。

（1）K-fold

K 折验证将除去测试集的剩余数据，切割为 K 份，每次将其中一个作为验证集，剩余的 K-1 个作为训练集，一共进行 K 次，将每次结果计算出平均数，即模型的性能效果。具体过程如图 2.4 所示：

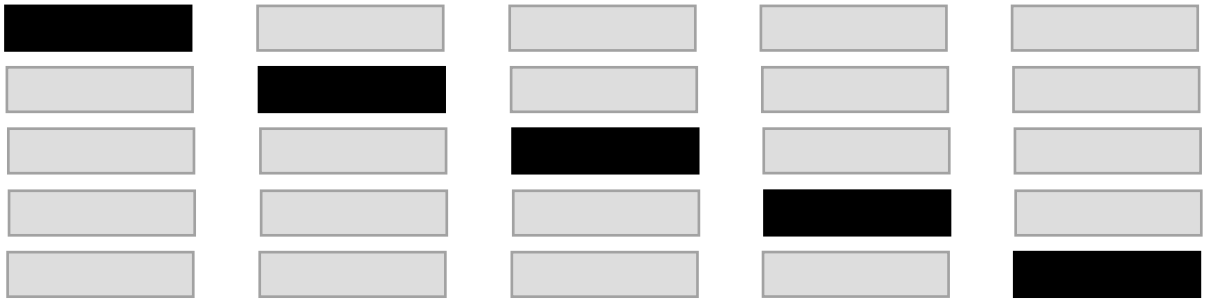

图 2.4 K-fold（五折）示意图

（2）Leave One Out

留一验证为 K-fold 验证的特殊情况，即每次只留出一条数据作为验证集，其余的部分都为训练集，直到所有的数据都曾作为验证数据。这一方法优点在于能够更加客观的评估模型，提高模型性能，缺点在于效率过低，对于大数据量的情形不适用。

第 3 章 相关特征指标选取

3.1 数据来源与时间范围

本文数据来源主要有两个：同花顺 iFinD 金融数据终端和中经网统计数据库。

本次测试中，相关的经济环境数据来自于中经网统计数据库的全国宏观年度库和分省宏观年度库，省份的经济环境数据是面板数据形式保存。由于目的是预测我国城投债的违约可能，所以只选取了全国 31 个省份的相关数据（不包括台湾省、香港特别行政区和澳门特别行政区）。经济统计数据类型的选取主要是依据费凡等人对于城投债信用利差的影响因素分析的综合结果[38]，其余有关特征指标的选取借鉴了文献[8-10]中的相关研究。并且根据数据预处理中数据核审的原则，从多个特征指标中进行筛选，去除劣质的、不完整的或是其他对于结果有负面影响的数据。同时，在收集数据过程中发现不同省份的地区经济数据公布时间不一致。在本文进行特征指标选取时，不少省份还未公布自身的经济统计数据，为了不增加噪声且确保本次建模的成功进行，本文没有选取 2020 年的经济数据，选取数据的时间范围为 2012 年至 2019 年的相关统计数据。

基于经济数据的限制，本文选取的债券数据时间跨度也是为 2016 年至 2020 年。以 2016 年为起始点是由于 2016 年后，国企、央企违约债券数量呈上升趋势，同时违约债券的发行主体评级也呈上升趋势，从原先的 B、C，到如今的 AAA。

根据以上因素，本文从同花顺 iFinD 金融数据终端获取地方融资平台发行债券以及国企、央企债券的相关数据。国企、央企债券的选取时间范围为 2016 年 1 月 1 日至 2021 年 1 月 1 日，债券种类为时间范围内到期、未到期或是发生违约情况的债券，其中，共有符合条件的国企、央企债券数据 71654 条，包含违约债券数量 142 条。城投债债券选取条件为 2020 年内到期或是未到期的债券，选取符合条件的城投债数据 14858 条。同时，为了避免未来数据对于结果的影响，每条债券数据匹配的宏观经济指标和地区经济指标为上一年的统计数据。

3.2 债券自身指标

债券自身指标一共选取了 9 个特征指标，分别为债券代码、债券是否违约、违约年份/当前年份、债券类型（THS 债券一级类型）、债券期限（年）、发行价格（元）、发行总额（亿）、债券发行时利息和债券起息日期。

其中债券代码作为整体数据表的主键，设为列标签；是否违约列的数据类型为字符串，分为“是”、“否”两类；债券违约日期或是当前日期数据类型为日期（date），利用 Python 进行切片处理，保留年份数据；债券类型的数据类型是字符串类型，依据同花顺债券一级分类标准：标准化票据、公司债、短期融资券、企业债、中期票据等；债券期限的数据类型为浮点类型，计量单位为年，保留四位小数；债券发行价格的数据类型为浮点类型，计量单位为元，保留四位小数；债券发行总额的数据类型为浮点类型，计量单位为亿元，保留四位小数；债券发行时利息的数据类型为浮点类型，计量单位为%，保留四位小数；债券起息日期的数据类型为日期（date），同时也利用 Python 进行切片处理，保留年份数据。具体如表 3.1 所示：

表 3.1 债券自身指标信息表

指标	类型	描述
Id	str	债券代码
Breach	str	是否违约
breachNewYear	date	违约年份/当前年份
typeBond	str	债券类型（THS 债券一级类型）
bondTerm	float	债券期限（年）
bondIssuePrice	float	发行价格（元）
bondTotalAmt	float	发行总额（亿）
bondInterest	float	债券发行时利息(%)
intBeginDate	date	债券起息日期

3.3 发债主体相关指标

债券发债主体的相关指标一共选取了 7 个特征指标，分别为发行主体所处地区（省）、发行主体债券余额（亿）、发行主体注册资金（亿）、发行主体公司规模、发行主体所处行业(GICS)、是否存在担保企业、发行主体发行时评级。

其中，发行主体所处地区根据同花顺 iFinD 金融数据终端可以定位到县级，但考虑到目前网上公开的经济数据在市一级已经存在大量缺失数据，且不同城市对于数据公开程度不一致，本文决定将债券所处省份作为分析的特征指标，数据类型为字符串类型。在后续改进中，如果能够获取到县一级的经济统计数据，理论上模型算法能够得到更为有效的结果。发行主体债券余额和注册资金的数据类型为浮点类型，计量单位为亿元，保留四位小数；发行主体公司规模的数据类型为字符串类型，分为“微型企业”、“小型企业”、“中型企业”和“大型企业”四类；发行主体所处行业的数据类型为字符串类型，分类依据是全球行业分类系统(GICS)一级类型：房地产、信息技术、金融、工业等；是否存在担保企业的数据类型为字符串类型，分为“是”、“否”两类；发行主体发行时评级的数据类型为字符串类型：BBB、BB、AA 等。具体如表 3.2 所示：

表 3.2 发债主体指标信息表

指标	类型	描述
Region	str	发行主体所处地区（省）
bondBalance	float	发行主体债券余额（亿）
regCap	float	发行主体注册资金（亿）
comScale	str	发行主体公司规模
Industry	str	发行主体所处行业(GICS)
guaCom	str	是否存在担保企业
bondIssuerRating	str	发行主体发行时评级

3.4 宏观指标

宏观经济统计数据共有 10 个特征指标，主要是将五种宏观经济指标与违约/当前年份和起息日期相匹配，旨在观察债券违约是否与宏观经济变动有关。选取的五种宏观经济指标参考了徐兆丰对于城投债信用风险影响因素的研究[39]，分别为：GDP、GDP 增长指数（以 2000 年 GDP 为 100）、社会融资规模存量增速（期末同比增速）、消费者水平和消费者水平指数（以 1978 年消费者指数为 100）。然后利用 pandas，根据每一条债券数据中的违约/当前年份和起息日期，匹配相应的宏观经济数据，并作为新的一列添加在数据表的末尾，将数据表保存为 CSV 文档以提高读写速度。其中数据类型均为浮点类型，具体如表 3.3 所示：

表 3.3 宏观指标信息表

指标	类型	描述
breachNewYearGDP	float	违约/当前年份国家 GDP
intBeginDateGDP	float	起息日期国家 GDP
breachNewYearGDPgrowthindex	float	违约/当前年份国家 GDP 增长指数
intBeginDateGDPgrowthindex	float	起息日期国家 GDP 增长指数
breachNewYearFinanceScaleRate	float	违约/当前年份社会融资规模存量增速
intBeginYearFinanceScaleRate	float	起息日期社会融资规模存量增速
breachNewYearComsumptionLevel	float	违约/当前年份消费者水平
intBeginYearComsumptionLevel	float	起息日期消费者水平
breachNewYear-ComsumptionLevelIndex	float	违约/当前年份消费者水平指数
intBeginYearComsumptionLevelIndex	float	起息日期消费者水平指数

3.5 区域指标

区域经济指标共选取了 15 个特征指标，分别为各个省份的 GDP、GDP 增长指数（以上一年 GDP 为 100）、消费者价格指数（以上一年 GDP 为 100）、常住人口自然增长率（‰）、人口密度（人/平方公里）、一般公共预算收入（亿元）、一般公共预算支出（亿元）、省级政府财政自给率（一般公共预算收入/一般公共预算支出）、城区面积（平方公里）、地级市数量、人均可支配收入（元）、电力消耗量（亿千瓦时）、商品房价格（元/平方米）、失业率（%）和最低生活保障人数。

数据来源是中经网统计数据库，数据保存形式为 15 个 CSV 格式面板数据，数据类型为浮点类型。由于每条数据的定位有两个，分别是违约/当前年份和发行主体所处地区，所以利用 pandas 的函数进行处理，将每个特征指标按照省份和违约/当前年份作为新的一列插入数据表的末尾。操作过程中，发现有些国企、央企所处的地区为香港特别行政区、澳门特别行政区等，考虑在这些地区不存在城投债，因此这类债券数据对本文研究目的无意义，并会增加模型噪声，综上考虑，故删去此类数据。最后将收集处理好的数据表保存为 CSV 格式，以待后续数据预处理使用。区域指标具体如表 3.4 所示：

表 3.4 区域指标信息表（省级）

指标	类型	描述
sbreachNewYearGDP	float	违约/当前年份 GDP
sbreachNewYear-GDPGrowthIndex	float	违约/当前年份 GDP 增长指数
sbreachNewYearCPI	float	违约/当前年份消费者价格指数
sbreachNewYearPNGT	float	违约/当前年份常住人口自然增长率
sbreachNewYearPD	float	违约/当前年份人口密度
sbreachNewYearGPBR	float	违约/当前年份一般公共预算收入
sbreachNewYearGPBE	float	违约/当前年份一般公共预算支出
sSSFR	float	政府财政自给率
sbreachNewYearUrbanArea	float	违约/当前年份城区面积
sbreachNewYearCityNumber	float	违约/当前年份地级市数量
sbreachNewYearPCDI	float	违约/当前年份-居民人均可支配收入
sbreachNewYearPowerUsage	float	违约/当前年份电力消耗量
sbreachNewYearHousePrice	float	违约/当前年份商品房价格
sbreachNewYear-UnemploymentRate	float	违约/当前年份失业率
sbreachNewYear-MinimumLivingSecurity	float	违约/当前年份最低生活保障人数

第 4 章 城投债信用风险预警模型的构建及优化

4.1 数据预处理

如上文所述，本文共初步筛选出符合条件的国企、央企债券数据 71654 条，其中包括违约数据 142 条；符合条件的城投债数据 14858 条。特征指标初步选取了可能有关的债券与经济指标共四十多个。数据来源是同花顺 iFinD 金融数据终端和中经网统计数据库。

这些初步收集的数据是“脏数据”，即为未经过处理的不完整的数据，存在大量的数据缺漏和不一致。这些数据一般无法直接使用或者可能导致输出结果不准确。为了确保模型的有效性，本文通过多个步骤：数据审核筛选、数据清理、数据集成、数据描述性分析、数据变换、数据归约对于收集到的数据进行预处理。最终选择的特征指标有 39 个，具体选取内容为上文所述的 39 个相关指标。

4.1.1 数据审核筛选

在上一章节中，本文进行了数据审核筛选步骤，具体内容为首先基于一致性的准则，为了使数据在国企、央企债券和地方融资平台债券两方面具有可比性和联系，同时为了方便后续的模型迁移学习，初步确定了四十多个国企央企债券和地方融资平台债券共有的特征指标并进行数据采集；其次，基于及时性的准则，根据数据的详细程度调整特征指标的选取，并删去部分缺失值较多或是没有及时更新数据的特征指标。然后基于适用性的准则，删去发行主体信用评级变化、发行主体最新信用评级和是否存在信用风险事件的指标。这是由于在实际收集中，发现这三个指标通常包含了未来数据，即违约事件的发生对这三个指标的内容产生直接影响，而通常难以将这些包含未来数据的债券数据手动筛选出来，所以决定删去这三个特征指标以保障模型结果的有效性。最后，删去债券发行主体的公司名称和担保企业的公司名称，原因是这两个指标对于结果无意义，本文的目的是为了探究地方融资平台债券违约与经济和自身指标的关系，故选择删去这两

个指标，避免数据噪声对结果的影响。

4.1.2 数据清理

首先，对于整体数据，本文以 id 为主键，删去重复值；并在 dataframe 中将 id 列设为整体的行标签，避免 id 列的数据也进入到建立模型的计算过程之中。其次，是对整体的缺失值进行处理，本次数据中的缺失值主要存在于债券自身指标和发债主体相关指标。

目前常见的缺失值处理手段有以下几种：1、替代法，即用其他数据或是邻近数据的中位数、均值等统计数据填补空值；2、数据删除法，即删去空值所在的数据行。在实际应用中由于不少模型允许空值的存在，所以也可以按照空值的数量来进行判断是否删去整行数据，例如利用 pandas 中的 thresh 参数进行判断，当一行的数据中非空值比 n 小时则删除该行数据（n 为自行设置的参数值）。3、变量删除法，即删除空值较多的特征指标，本文在选取区域经济指标和宏观经济指标时运用该方法进行特征指标的筛选。

在本次步骤中主要利用数据删除法删除此前筛选出的 39 个特征指标中的空值，由于后续需要对多个机器学习算法模型在本问题中的优劣进行比较，其中部分算法模型不允许数据中存在空值，所以本次数据清洗中将全部空值的所在行数据直接进行删除处理。经过处理后的数据为国企、央企债券数据 62647 条，其中违约数据 72 条；地方融资平台债券数据 13964 条。

4.1.3 数据集成

本步骤中，以 id 为连接键将债券自身指标、发债主体相关指标这两个数据表进行合并；然后以省份和年份（前一年）作为定位条件合并地区经济指标、宏观经济指标；最后将整合后的国企、央企债券数据表命名为 test2.0，地方融资平台债券命名为 ctest2.0，以便后续的分析调用。

4.1.4 数据描述性分析

利用 pandas 的 describe 函数对于两个数据表的部分特征指标进行简单的描述性分析：

（1） 国企、央企债券数据：

1、发行主体债券余额分析：

表 4.1 bondBalance 特征描述表

bondBalance/发行主体债券余额（亿）	
count	62647
mean	2230.267537
std	3401.400396
min	0.000000
25%	125.800000
50%	408.417864
75%	2177.000000
max	86951.396040

如表 4.1 所示，发债主体的平均债券余额约为 2230 亿元，标准差约为 3401。可以看出不同的国企、央企债券余额相差较大。同时 25%百分位数约为 125 亿元，50%百分位数约为 408 亿元，75%百分位数约为 2177 亿元，整体数据中有高于 75%企业的债券余额不足平均余额，结合最大值数据，即企业最多有 86951 亿元的负债，为均值的 40 倍左右，以上数据说明企业负债情况两极分化严重。

2、发行主体注册资金分析：

表 4.2 regCap 特征描述表

regCap/发行主体注册资金（亿）	
count	62647
mean	307.809451
std	1155.002734
min	0.080000
25%	37.759450
50%	77.504314
75%	196.871963
max	17395.000000

如表 4.2 所示，国企、央企注册资金的总体分布特点和其债券余额有着相似的性质，同样是两级分化，75%百分位数远小于注册资金的均值，最大值为 17395 亿元，利用 SPSS 进行相关性检验，债券余额和注册资金的 Spearman 相关性检验数值为 0.835，为强相关。说明目前的国企、央企债券市场上存在大企业，这些大企业也吸收了最多的资金，同时大部分的债券数量还是由中小企业贡献。

3、债券期限分析：

表 4.3 bondTerm 特征描述表

bondTerm/债券期限	
count	62647
mean	0.965660
std	1.551403
min	0.019200
25%	0.251400
50%	0.504100
75%	1.000000
max	30.000000

如表 4.3 所示，国企、央企债券平均期限仅为 0.965 年，其中 75%百分位数为 1，由此可知约有四分之三的债券为短期债券，剩下的为中长期债券，最高的债券期限为 30 年。正如前文分析，最近几年的债券违约事件使得投资者更倾向于短期债券以减少损失，同时，市场也相应更多地发行短期债券来满足投资者的偏好。故市场中存在着大量的短期债券。

4、发行总额分析：

表 4.4 bondTotalAmt 特征描述表

bondTotalAmt/发行总额（亿）	
count	62647
mean	12.060251
std	25.707529
min	0.100000
25%	1.500000
50%	5.000000
75%	12.000000
max	850.000000

如表 4.4 所示，整体分布特征和发行主体债券余额相似，75%百分位数与平均值接近约为 12 亿元，而 50%百分位数仅为 5 亿元，最大值为 850 亿元。以上数据说明，单个债券的分布情况与企业整体债券余额情况相关，利用 SPSS 进行相关性检验，两者的 Spearman 相关性检验数值为 0.243，为弱相关。

（2） 地方融资平台债券数据：

1、发行主体债券余额分析：

表 4.5 bondBalance 特征描述表

bondBalance/发行主体债券余额（亿）	
count	13964
mean	121.075590
std	196.832459
min	0.000000
25%	26.000000
50%	63.000000
75%	131.430000
max	1599.451244

如表 4.5 所示，发债主体的平均债券余额约为 121 亿元，标准差约为 196。75%百分位数为 131 亿元，略大于平均值。整体的数值比例分布和国企、央企债券相似，但是在债券余额数值上远小于国企、央企债券，最大值仅为 1599 亿元。可能的原因是城投债的发债主体大多数是地方性的，不像国企、央企中可能存在全国性的企业，所以两者的资本和负债数值不同。

2、发行主体注册资金分析：

表 4.6 regCap 特征描述表

regCap/发行主体注册资金（亿）	
count	13964
mean	62.531973
std	138.642799
min	0.100000
25%	8.000000
50%	20.000000
75%	50.000000
max	1452.905491

如表 4.6 所示，相比于国企、央企债券的数据，城投债的数据离散程度较低。城投债的发债主体—地方融资平台的平均注册资金约为 63 亿元，最小注册资金为 1000 万元，最大为 1453 亿元，25%百分位数为 8 亿元，75%百分位数为 50 亿元，小于平均数。根据地方融资平台数据，债券余额和注册资金的 Spearman 相关性检验数值为 0.552，为中相关。

3、债券期限分析：

表 4.7 bondTerm 特征描述表

bondTerm/债券期限	
count	13964
mean	5.001374
std	2.237545
min	0.164400
25%	3.000000
50%	5.000000
75%	7.000000
max	28.000000

如表 4.7 所示，在债券期限方面，城投债和国企、央企债券有很大的区别。相比于国企、央企债券的以短期债券为主，城投债更多的是中期债券。从 25%百分位数为 3 年，75%百分位数为 7 年，可以得到大部分的城投债均为中期债券，平均的债券期限也为五年，远远大于国企、央企债券的一年。但是在最长债券期限中，城投债小于国企、央企数据，最大值为 28 年。

4、发行总额分析：

表 4.8 bondTotalAmt 特征描述表

bondTotalAmt/发行总额（亿）	
count	13964
mean	9.040980
std	5.955683
min	0.000000
25%	5.000000
50%	8.000000
75%	10.000000
max	82.000000

如表 4.8 所示，城投债平均值约为 9 亿元。最大值为 82 亿元，最小值为 0。其中最小值为 0 的可能原因是单次债券总额较小或是数据不准确而产生的噪音。其余指标中，25%百分位数为 5 亿元，50%百分位数为 8 亿元，75%百分位数为 10 亿元。同样使用 SPSS，对于发债主体债券余额和发行总额进行相关性检验，得到的 Spearman 相关性检验数值为 0.321，为弱相关。

4.1.5 数据变换

（1）非数值类型数据处理：

此时，两个数据表中的初步数据处理已经完成，但依旧不能够直接建立模型进行分析。通过 Python 库的提示发现，不少模型不支持对于文本数据和日期数据进行学习。本次使用文本、日期数据目的较为简单，是对数据的分类，所以直接使用 pandas 库自带的非数值类型数据处理函数 replace 对于文字用随机数字进行替换。为了避免数据不一致，首先对两个数据表纵向进行合并，然后利用 replace 函数对于非数值类型数据进行处理，使相同的文字数据替换的数字一致。最后按照国企、央企和地方融资平台两个不同的发债主体企业性质对整张数据表进行分割，得到国企、央企债券数据表 test2.1 和城投债数据表 ctest2.1。

（2）数据标准化：

随着机器学习和多指标模型的发展，数据标准化也逐渐成为数据处理中的一个重要步骤。由于特征指标之间存在差异，并且通常情况下，不同单位的指标数值相差极大。如果不经过数据标准化处理，部分具有高数值的特征会在起初具有较高的权重，同时低数值的特征的作用性会被削弱，所以为了避免数据本身特点影响结果，提高结果的可靠性，需要对于之前处理后的数据进行标准化处理。

数据标准化即将数据按照比例或是一定算法变为一个特定区间内的数据。通过这种方式，去除了特征指标之间的单位差异。标准化的基础公式为：

$$x_i^* = (x_i - mean)/std \qquad (4\text{-}1)$$

(mean 为每列数据的均值，std 为每列数据的标准差)。

在机器学习中，概率模型（如决策树等）不需要进行标准化，因为特征变量的值对结果没有影响，更注重的是特征变量的分布和彼此间的条件概率；而 Adaboost、SVM、KNN 等最优化问题需要进行标准化处理。为了后续比较不同模型对于数据分析结果优劣的需要，本文利用 sklearn 库中的 StandardScaler 函数对数据进行标准化，使每个特征数据的均值为 0，方差为 1；并且保留了数据在各个维度上的信息。

在标准化处理中，通常有以下方法，一是不分割训练集和测试集，利用整体数据的均值和标准差进行标准化处理；二是利用训练集的均值和标准差对训练集和测试集进行标准化操作。在实际应用环节中，由于测试集代表的是未来数据，所以第二种方法更为合理。

StandardScaler 函数提供了保存训练集的均值、标准差来转换测试集的功能，故在标准化之前，需要对数据表进行切割。首先，将目标变量（breach）和特征变量单独提取，分别命名为 y 和 X，其次使用 train_test_split 函数随机将样本数据 X 分割为训练集和测试集，分割比例 test_size 为 0.2，即测试集数据量占整体样本数据量的 20%，超参数 random_state 为默认，即每次采样都是随机抽取 20%。然后用 StandardScaler 函数中的 fit_transform 对于训练集数据进行先拟合后标准化操作，此时 StandardScaler 函数中便保存了训练集的均值和标准差。之后通过 transform 功能，基于训练集的参数对测试集进

行标准化操作，模拟实际应用中的情况。

4.1.6 数据归约

数据归约即在保留数据所需原始信息的前提下，尽可能地精简数据量。常见的数据归约方法有维归约（小波变换、主成分分析法）、数量归约和特征值归约（特征值离散化技术）。维归约是通过减少弱相关的数据或是重新构造的方式减少特征变量的数量；数量归约是通过一定的方式从样本总体中抽取具有代表性的数据子集，此方法的优点是能够减少计算成本，提高求解速度，有可能达到更高的精度，缺点是由于取样误差的存在，子集和原始样本之间存在差异，当子集越大时，差异越小；特征值归约对于连续的特征变量值进行离散化处理，使大量的具体数据变换为少量的区间。特征值归约分为有参、无参两种，有参的可以分为回归方法和对数线性模型，无参的可以分为直方图、聚类等。数据归约可以避免过高维度导致的机器学习算法失效，同时加快运算速度，减少后续的机器算法比较和超参数优化的时间。

本文中利用主成分分析法进行数据归约，使用 python 导入数据，然后选取了 sklearn.decomposition.PCA 函数进行降维，这是一个封装好的工具库，调用后通常不需要修改函数内部参数，常见的方法是操纵 n_components 这参数对输出维度按照所需进行调整。n_components 的取值可以有三种方式，第一种为大于 1 的正整数，即直接指定所要降维到的维数；第二种为（0,1]，即代表指定主成分的方差和所占的最小比例阈值；第三种方式为指定方法，例如‘mle’，即模型利用极大似然估计算法根据特征数据的方差特点进行降维维数的选择，默认的方法为‘min’。

其余的参数有 whiten，即是否白化，也就是数据标准化，由于本文在此前已经进行了数据标准化操作，所以使用它的默认值：False。svd_solver 是对奇异值分解（SVD）方法的选择，有‘auto’、 ‘full’、‘arpack’、‘randomized’四种选择。默认值为‘auto’，即由算法在后面的三种方法中自主选择合适的方法进行降维。

本次仅操作 n_components 一个参数，将其设置为 0.95，即主成分的方差和所占的最小比例阈值为 0.95，保留了 95%的信息。利用训练集数据进行拟合后，得到降维后的数据为 18 维。利用 explained_variance_ratio_观察经过降维处理后的各个主成分的方差值

占总方差值的比例。比例数值越高，该主成分的重要性越大，具体数值如表 4.9 所示：

表 4.9 降维后新变量方差占比表

新变量名	占总方差值的比例
主成分 1	18.854850%
主成分 2	15.378397%
主成分 3	13.709736%
主成分 4	12.252201%
主成分 5	5.489498%
主成分 6	4.298845%
主成分 7	3.830883%
主成分 8	3.186244%
主成分 9	2.954018%
主成分 10	2.318876%
主成分 11	2.232182%
主成分 12	1.985515%
主成分 13	1.884163%
主成分 14	1.607342%
主成分 15	1.493226%
主成分 16	1.412398%
主成分 17	1.238617%
主成分 18	1.048369%

从表 4.9 中可以看出，各个主成分重要性分布较为均匀，不存在某个或是少数主成分变量就代表了整体数据特征的现象，每一个主成分都有一定的重要性。最后用 transform 功能，根据训练集的数据拟合结果对测试集进行降维。并将特征降维的系数矩阵保存为 CSV 格式，命名为 PCA.csv，详细数据内容参见附录。至此，数据预处理已经完成，下一步进行机器学习算法模型的选取和优化。

4.2 模型比较及选取—XGBoost 模型

4.2.1 模型比较

为了得到更好的预测效果，本文用几种常见的机器学习算法模型对于样本数据集进行学习。随着机器学习领域的发展，目前可用于分类的机器学习算法数量众多，改良后的算法或是组合优化后的算法更是层出不穷，它们各有各的优缺点，适用的范围也有所不同。本文选取了八个常见的机器学习算法：XGBoost 模型、随机森林模型、ET 模型、高斯朴素贝叶斯模型、K 最邻近模型、逻辑回归模型、决策树模型和支持向量机模型，在对这八个模型不调优超参数的前提下，用默认的模型超参数组合对数据集进行学习，观察它们的分类效果。

为了更好地比较结果的区别，本文将 ROC 曲线的 AUC 值作为判断模型优劣的标准。ROC 曲线为接收者操作特征曲线，来自于二战中检测敌方载具准确性的指标，可以用于区别信号和噪声。后来被广泛应用于机器学习中分类模型的评判中，ROC 曲线的横轴为假正率（FPR：分类为正，实际为负），纵轴为真正率（TPR：分类为正，实际为正）。根据二分类模型中阈值的不同，模型结果会产生不同的假正率和真正率，所有阈值产生的点连接后便是 ROC 曲线。AUC 值便是 ROC 曲线下方面积，AUC 值越接近于 1，说明模型的分类结果越好。

同时，为了避免模型结果的偶然性，同时由于留一交叉验证法耗时过长，故选用目前主流使用的 K-fold 交叉验证法减少噪声和随机性的影响，K 值设定为 10。使用 sklearn 库中的 cross_val_score 交叉验证评估函数进行模型评估，其中的 scoring 参数设置为‘roc_auc’，即以 AUC 值作为评分标准。表 4.10 中为八种机器学习算法模型的 AUC 值：

表 4.10 常见机器学习算法模型 AUC 值

模型名称	AUC 值
XGBoost 模型	0.991727
随机森林模型	0.943631
ET 模型	0.824924
高斯朴素贝叶斯模型	0.909468
K 最邻近模型	0.910092
逻辑回归模型	0.974354
决策树模型	0.775503
支持向量机模型	0.929973

根据表 4.10 数据显示，在 10 次交叉验证的均值中，XGBoost 算法模型的结果明显优于其余七种算法模型，所以本文选择 XGBoost 算法用于本次城投债违约风险预警模型构建。

4.2.2 XGBoost 算法原理

XGBoost（极端梯度提升算法）算法是 Boosting（提升）算法的一种，也是基于 GDBT（梯度提升算法）的优化算法。GDBT 是利用 Boosting 理论的加法算法，即在模型训练时使用前向分布算法进行贪婪学习，每次只学习一个基学习器，逐渐优化损失函数。它迭代时，学习一个 CART 树来拟合前面全部决策树的预测与实际值的残差（利用负梯度模拟残差）。XGBoost 算法首次于 2016 年被提出，其基本原理和 GDBT 相同，如下所示：

$$\hat{y}_i^{(0)} = 0 \tag{4-2}$$

$$\hat{y}_i^{(1)} = f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i) \tag{4-3}$$

$$\hat{y}_i^{(2)} = f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \tag{4-4}$$

……

$$\hat{y}_i^{(t)} = \sum_{k-1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i) \tag{4-5}$$

以上式子模拟 XGBoost 的迭代过程，$\hat{y}_i^{(t)}$为第 t 轮的模型预测，$\hat{y}_i^{(t-1)}$为此前 t-1 轮的模型预测结果，$f_t(x_i)$为新增的基学习器。

XGBoost 在 GDBT 的基础上进行了不少优化：1、使用泰勒公式的二阶导，从而提高精度，减少误差；2、使用正则项对模型进行简化，避免过拟合；3、可以并行运算，大大提高了运算速度。4、剪枝优化，不同于 GDBT 在遇到负损失后停止分裂，XGBoost 持续分裂到指定最大深度后再决定是否剪枝，这使得整体为正损失（负损失后存在更大的正损失）的子节点得以保留

机器学习的目标函数一般为以下格式：

$$Obj(\theta) = L(\theta) + O(\theta) \tag{4-6}$$

其中，$L(\theta)$表示模型的损失函数，$O(\theta)$表示模型的复杂度函数。机器学习的目标是使$Obj(\theta)$最小，即损失函数值较小时，模型的复杂度也同样较低，此时便为最优解。在实际应用中，通常模型的复杂度随着损失函数值的下降而增加，当复杂度过高时，会出现过拟合现象，即模型只在当前数据集表现良好；而复杂度过低时，则会出现欠拟合现象，模型结果的误差较大。

XGBoost 算法模型的目标函数为（以平方损失函数为例）：

$$Obj^t = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) + constant \tag{4-7}$$

其中，l代表损失函数$l\left(y_i, \hat{y}_i^{(t-1)}\right) = (y_i - \hat{y}_i^{(t-1)})^2$，$\Omega(f_t)$为惩罚项$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2$，T 为决策树叶子节点个数，$\frac{1}{2}\sum_{j=1}^{T} \omega_j^2$为权重的 L2 正则惩罚项，$\gamma$、$\lambda$为系数。带入计算：

$$Obj^t = \sum_{i=1}^{n} \left(y_i - \left(\hat{y}_i^{(t-1)} + f_t(x_i)\right)\right)^2 + \Omega(f_t) + constant \tag{4-8}$$

$$= \sum_{i=1}^{n} [2\left(\hat{y}_i^{(t-1)} - y_i\right) f_t(x_i) + f_t(x_i)^2] + \Omega(f_t) + constant \tag{4-9}$$

然后利用泰勒展开的二阶导近似目标：

$f(x)$泰勒二阶展开：

$$f(x + \triangle x) \simeq f(x) + f'(x) \triangle x + \frac{1}{2} f''(x) \triangle x^2 \tag{4-10}$$

定义：$g_i = \partial_{\hat{y}^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right)$，$h_i = \partial^2_{\hat{y}^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right)$　　（4-11）

$$Obj^t \simeq \sum_{i=1}^{n} [l\left(y_i, \hat{y}_i^{(t-1)}\right) + g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2] + \Omega(f_t) + constant \quad （4-12）$$

$$Obj^t \simeq \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2] + \Omega(f_t) + constant \quad （4-13）$$

其中，$l\left(y_i, \hat{y}_i^{(t-1)}\right)$相当于常数项，合并至末尾$constant$中。其次，展开惩罚项，由样本遍历转换为叶子节点遍历：

$$Obj^t \simeq \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2 \quad （4-14）$$

$$= \sum_{j=1}^{T} [g_i \omega_{q(x_i)} + \frac{1}{2} h_i \omega^2_{q(x_i)}] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2 \quad （4-15）$$

$$= \sum_{j=1}^{T} [(\sum_{i \in I_j} g_i)\omega_j + \frac{1}{2}(\sum_{i \in I_j} h_i + \lambda)\omega_j^2] + \gamma T \quad （4-16）$$

4.2.3 XGBoost 算法库常用超参数介绍

在机器学习中，超参数是指在训练模型之前就预先设定好的参数，而参数则是指模型通过对数据拟合得到的参数数据。超参数通常用于调整模型的学习方法，速度等方面变量。为了得到一个高效准确的模型结果、提高模型的准确率，超参数的设置至关重要。在 XGBoost 算法库中，tree boost 的常用超参数及意义如表 4.11 所示：

表 4.11 XGBoost 常用超参数表

超参数名称	介绍
max_depth	默认值为 6。用于调整 CART 树的最大深度，max_depth 取值越大，样本数据学习更为具体，也更有可能过拟合。
learning_rate	默认值 0.3。即每次迭代中利用之前数据误差的比例，取值越大，学习速度越快，但有概率出现梯度爆炸现象，使模型不稳定；取值过小，则不仅学习时间长，同时会出现过拟合现象。
n_estimators	默认值为 100。代表模型中树的数量，也就是模型迭代的总次数。当迭代次数过多时，容易引起过拟合，而次数过少时，则不能充分学习数据中的特征

续表 4.11

min_child_weight	默认值为 1。它代表叶子节点的权重之和。通常，这个超参数用于避免过拟合，可以通过提高取值的方法，阻止模型学习过于具体特殊的数据特征。其具体原理是当子节点的所有样本权重和大于 min_child_weight 值时便停止分裂。
max_delta_step	默认值为 0。用于限制决策树权重改变的最大范围，当取值为 0，即没有限制；当取值为正数时，可以对模型加以限制，使其在数据学习上更加保守。max_delta_step 在类极度不平衡的数据集中有一定的作用。
subsample	默认值为 1。是每次迭代中随机选取后的子样本的比例。当 subsample=0.6 时，表示迭代时使用 60%的训练数据。可以通过调低此超参数的方式来避免过拟合现象的出现。
colsample_bytree	默认值为 1。代表每次迭代时，子样本选取每个特征的数据占比。调低此超参数可以避免过拟合，但过小时会导致欠拟合。
reg_alpha	默认值为 0。即损失函数中 L1 正则惩罚项的系数。取值越大，惩罚力度越大，模型更加保守。
reg_lambda	默认值为 1。即损失函数中 L2 正则惩罚项的系数。取值越大，惩罚力度越大，模型更加保守。
scale_pos_weight	默认值为 1。此超参数对于类不平衡的二分类模型优化有很大帮助，能使数据集不平衡时的模型收敛。原理是通过对于少数类的数据进行过采样以平衡比例[40]。
gamma	默认值为 0。是决策树进行分裂时最小的损失函数损失值。当取值越大，说明较小的损失函数值下降将被模型忽视，即决策树分裂变少，模型变得更加保守。

4.3 模型建立及优化

4.3.1 模型构建

通常，算法库中的超参数都有默认值，适用于绝大多数情况。但这在确保泛用性的同时不免减少了对具体数据样本输出结果的准确度，为了得到更好的模型结果，需要对于超参数进行调整。不同的超参数组合对应决定了不同的模型输出结果，并且结果往往是连续、有迹可循的，所以可以利用调整超参数组合的方法逐渐逼近最优解。

首先，本文先手动调整超参数以初步构建 XGBoost 算法模型。定义损失函数 objective 为“binary:logistic”：这是算法库中内置用于学习二分类问题的二项逻辑回归方法，输出结果为概率；并将评分方法设置为 AUC 值；其余超参数设置如表 4.12 所示：

表 4.12 XGBoost 超参数取值表

超参数名称	取值
max_depth	5
learning_rate	0.1
n_estimators	500
min_child_weight	1
max_delta_step	0
subsample	0.8
colsample_bytree	0.8
reg_alpha	0
reg_lambda	1
scale_pos_weight	1
gamma	0

其次，使用经过降维后的训练训练集数据进行学习迭代，得到训练好的算法模型，将其命名为 clf_pca。然后使用 predict 函数，基于测试集数据，对于测试集的违约情况进行预测。最后，创建新的数据表，插入两列数据，第一列数据为测试集数据的预测值，另一列数据为测试集数据的实际值，得到模型预测结果的准确率为 99.920%（预测正确个数/总个数）。算法模型的 ROC 曲线如图 4.1 所示，其中 AUC 值 0.99941：

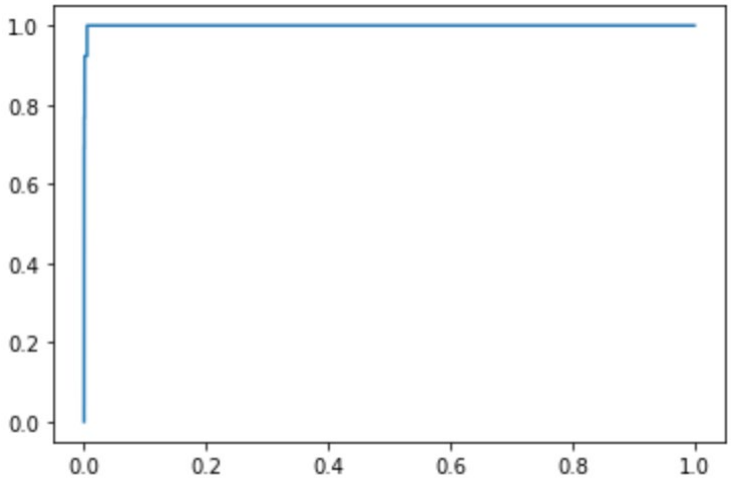

图 4.1 手动调参模型 ROC 曲线图

可以看出仅仅是经过初步训练后，XGBoost 算法便取得了良好的效果。为了分析不同特征指标对于模型的影响，利用 feature_importances_函数查看各个特征指标的重要性，具体结果如表 4.13 所示：

表 4.13 降维后特征指标重要性表

降维后变量名	重要性值
主成分 1	0.06254499
主成分 2	0.12491626
主成分 3	0.05059753
主成分 4	0.04082454
主成分 5	0.08624224
主成分 6	0.0531567
主成分 7	0.06143051
主成分 8	0.05328562
主成分 9	0.04390173
主成分 10	0.06456066
主成分 11	0.05403547
主成分 12	0.02597753
主成分 13	0.05008157
主成分 14	0.04010993
主成分 15	0.03532023
主成分 16	0.04448474
主成分 17	0.06372183
主成分 18	0.04480788

由于经过主成分分析法降维之后，每一个主成分特征变量是前文选择的 39 个原始特征变量的组合。它们的实际意义模糊、不直观，难以对其进行解释，因此无法有效地探究单个原始特征变量的重要性影响。为了得到原始特征指标对于模型的直接影响和为后续的进一步研究提供帮助，本文使用未降维的数据再次进行学习建模并对测试集进行预测，得到各个特征指标在新训练的模型中重要性数值，并对其按大小顺序进行排序，结果如表 4.14 所示：

表 4.14 原始特征指标重要性表

特征指标名称	重要性值
breachNewYear	0.290492
breachNewYearGDP	0.165069
breachNewYearGDPgrowthindex	0.059685
bondIssuePrice	0.057451
bondBalance	0.043829
regCap	0.024492
bondIssuerRating	0.023508
sbreachNewYearMinimumLivingSecurity	0.023349
sbreachNewYearUnemploymentRate	0.020253
region	0.020009
・・・	・・・
breachNewYearConsumptionLevelIndex	0.000000
breachNewYearConsumptionLevel	0.000000
breachNewYearFinanceScaleRate	0.000000
intBeginDateConsumptionLevelIndex	0.000000

由于共有 39 个特征变量，本文并未在此列出完整表格，具体表格数据内容在附录中。上表共选取 14 个有代表性的特征指标重要性值，包括前十个对模型影响最大的特征指标和末尾四个具有特殊性质的指标。

据上表可以看出对于分类结果最重要的前十个特征分别是：违约/当前年份、违约/当前年份国家 GDP、违约/当前年份国家 GDP 增长指数、发行价格（元）、发行主体债券余额（亿）、发行主体注册资金（亿）、发行主体发行时评级、违约/当前年份最低生活保障人数、违约/当前年份失业率、发行主体所处地区（省）。除去违约/当前年份和发行主体所处地区（省）这两个稍有特殊的特征指标，其中于债券自身或发债主体有关的指标占 50%，经济数据指标占 50%（宏观数据占 25%，省级数据占 25%），恰好符合前文

的四个板块分类，说明此前的版块划分有一定合理性。同时，观察到违约/当前年份和当时的国家 GDP 数值相比其余八个有明显优势，可能的原因是目前债券违约数量呈逐年上升的趋势且违约债券数量整体不多，所以分类结果受这两个特征影响较大。

根据上表，尾部的四个特征变量数值具有一定的特殊性，它们对模型分类结果的影响近乎于零。这四个特征指标具体分别是：违约/当前年份消费者水平指数、违约/当前年份消费者水平、违约/当前年份社会融资规模存量增速和起息日期消费者水平指数。在后续的进一步研究中可以考虑删去这四个指标或避免增添和这四个指标联系较大的新特征指标。

经过实际检验，新构建的模型准确率也有着良好的结果，为 99.912%。但和使用降维后数据的模型准确率依旧有一定的差距，这说明主成分分析法降维对于模型结果有一定的改善效果。所以，后续选用降维后的数据进行优化调试。

4.3.2 模型优化

前文之中，初步构建的 XGBoost 算法模型已经证明其可行性，接下来将对它进行调参优化。机器学习中模型的超参数优化通常是黑盒过程，即只能得到输入值和输出值，所以只能通过不断试错的方式，寻找最优超参数组合。通过学习前人文献[49]，发现除手动调参以外，自动调参也可以细分为逐步调参和整体调参，逐步调参的优点在于每步调参的所需时间较短，可以在更大取值范围内，寻找单个超参数的最优取值。它的缺点在于需要对于数据整体和模型常用超参数设置有深度的理解，且由于是分步寻找最优值，每次只调整一个或两个超参数，就有可能陷入局部最优值的困境，错过了更好的全局最优值。而整体调优的优点是可以在高维的超参数组合中寻找最优解，一定程度上避免了得到局部最优解的可能，缺点则是搜寻最优解的耗时较长。目前常见的自动调优算法有：网格搜索法、随机网格搜索法、二分法和贝叶斯优化，其中二分法仅适用于超参数逐步调优，而其他三种方法对于逐步调优和整体调优均适用。本文选取整体调优的方法，利用机器学习领域中最为广泛使用的三种自动调参算法结合交叉验证法对其进行超参数组合优化：

（1） 网格搜索法

正如第二章所说，网格搜索法属于穷举算法，每增加一个超参数便会导致耗时指数级的增长。本文首先利用网格搜索法，尝试对于 11 个常用的超参数进行调优。发现在耗时 48 小时后，依旧没有完成本轮调优。于是考虑简化网格搜索中的超参数数量，设置为仅对三个相对更重要的超参数进行调优，分别为：*max_depth*、*learning_rate*、*n_estimators*，取值范围为固定的数值组合，具体超参数设置如表 4.15 所示：

表 4.15 网格搜索法超参数表

超参数名称	取值
max_depth	[1, 2, 5, 10]
learning_rate	[0.02, 0.05, 0.15, 0.2]
n_estimators	[300, 500, 700]

具体步骤为：在 Python 中，导入 GridSearchCV 函数，该函数将网格搜索法与交叉验证法结合，帮助研究者更为方便地使用网格搜索法。其次将所用的算法模型和超参数的取值组合导入 GridSearchCV 函数，并设置为 10 折交叉组合，将评分标准设置为 AUC 值。运行函数后，得到在取值范围内最优的超参数组合是：*max_depth* = 10、*learning_rate* = 0.05、*n_estimators* = 300，即决策树最大深度为 10，学习率为 0.05，迭代次数设置为 300 次（300 个基学习器）。本次三个超参数组合调优共耗时 1 小时左右。代入函数给出的最优超参数组合后，10 折交叉验证的 AUC 均值为 0.993785。

（2） 随机网格搜索法

由于普通网格搜索法耗时过长，所以本文又尝试使用随机网格搜索法对超参数组合进行调优。随机网格搜索法的特点是不按照超参数网格的顺序进行学习，而是通过分割超参数网格，然后随机取点进行计算，然后在较优值旁继续随机取点。这种方式大大减少了所需时间，缺点也是显而易见，由于改变了原有的穷举方法，随机网格搜索法有一定可能错过最优解，只能得到局部最优值。本文在这次调优中，设置超参数组合的取值范围如表 4.16 所示：

表 4.16 随机网格搜索法超参数表

超参数名称	取值
max_depth	[1, 2, 3, 5, 10]
learning_rate	[0.02, 0.05, 0.1, 0.15, 0.2]
n_estimators	[100,200, 300, 500, 700]
min_child_weight	[0, 2, 5, 10, 20]
max_delta_step	[0, 0.2, 0.6, 1, 2]
subsample	[0.6, 0.7, 0.8, 0.85, 0.95]
colsample_bytree	[0.5, 0.6, 0.7, 0.8, 0.9]
reg_alpha	[0, 0.25, 0.5, 0.75, 1]
reg_lambda	[0.2, 0.4, 0.6, 0.8, 1]
scale_pos_weight	[0.2, 0.4, 0.6, 0.8, 1]
gamma	[0,0.01,0.05,0.1,0.2,0.4,0.6]

具体步骤为：导入 RandomizedSearchCV 函数，函数与此前的 GridSearchCV 函数相似，只是由普通网格搜索法变为随机网格搜索法，函数同样内置了交叉验证法。其次将 XGBoost 算法模型和超参数取值组合输入函数之中，设置交叉验证次数为 10，评分标准为 AUC 值，得到最终的优化结果为最高 AUC 均值为 0.992598，与之对应的超参数组合为：*subsample* = 0.7、*scale_pos_weigh*t = 0.8、*reg_lambda* =1、*reg_alpha* =0.25、*n_estimators* = 300、 *min_child_weight* = 2、*max_depth* = 10、*max_delta_step* = 2、*learning_rate* = 0.15、*gamma* = 0.6、*colsample_bytree* = 0.7。其中，超参数的具体涵义参见上文中表 4.11。根据结果，可以看出这次超参数优化在 AUC 均值上小于此前的网格搜索法，出现此现象的原因可能是随机网格搜索法陷入了局部最优的困境，也有可能是随机采样数量不足的原因。但增加采样数量，依旧会出现优化过程时间过长的问题。本次调优时间大致为 30 分钟。虽然，耗费时间大大减少了，可输出结果未能达到预期效果。于是，尝试新的调优方法——贝叶斯优化。

（3） 贝叶斯优化

前文数据显示，两次调优的结果显示算法模型依旧存在一定的提升空间，于是本文尝试使用贝叶斯优化。在部分情况下，贝叶斯优化相对其他两种自动调优算法更为有效，并在一定机率上能够超越调参工程师的手动调参组合。它使得普通机器学习使用者也能够得到一个满意的结果。具体算法原理参见第二章的介绍，此处直接导入贝叶斯优化算法库进行超参数组合优化。相比于此前两次优化过程中超参数取值为有限的给定值，本次优化中将超参数取值设定为取值区间，从而给予贝叶斯优化算法更大的自由空间来优化结果，具体取值范围如表 4.17 所示：

表 4.17 贝叶斯优化超参数取值范围表

超参数名称	数据类型	取值范围
max_depth	int	1~10
learning_rate	float	0.02~ 0.2
n_estimators	int	200~1500
min_child_weight	int	0~20
max_delta_step	float	0.01~2
subsample	float	0.6~1
colsample_bytree	float	0.5~1
reg_alpha	float	0.01~1
reg_lambda	float	0.2~1
scale_pos_weight	float	0.2~1
gamma	float	0~0.5

其次，将超参数取值范围和 XGBoost 算法模型输入贝叶斯优化函数之中，设置其中的交叉验证为 10 次，评分标准为 AUC 值。根据实际情况，函数进行了多次尝试计算以寻找最优组合。在输出结果前，贝叶斯优化函数整体一共进行了 30 次组合尝试。具体计算过程、每次构建模型的 AUC 值和尝试的超参数组合如图 4.2 所示：

iter	target	colsam...	gamma	learni...	max_de...	max_depth	min_ch...	n_esti...	reg_alpha	reg_la...	scale_...	subsample
1	0.9608	0.8523	0.2344	0.1974	1.068	2.634	17.88	262.6	0.9483	0.9097	0.6294	0.7491
2	0.992	0.9403	0.4795	0.0847	1.713	7.969	0.4048	1.041e+0	0.6348	0.8973	0.7926	0.7758
3	0.948	0.9151	0.4201	0.02229	0.6887	6.729	12.0	865.2	0.9583	0.7423	0.4905	0.9629
4	0.9898	0.5199	0.2852	0.1128	1.212	3.546	5.89	1.345e+0	0.9527	0.5085	0.7515	0.8099
5	0.9922	0.7532	0.02569	0.09555	1.588	4.59	3.365	1.066e+0	0.1453	0.5499	0.8407	0.8187
6	0.9838	0.8092	0.2529	0.1409	2.0	2.129	0.04958	1.065e+0	0.01	0.5881	1.0	0.6611
7	0.951	0.6045	0.2305	0.05005	1.422	9.411	15.43	1.058e+0	0.06379	0.445	0.7182	0.891
8	0.9926	0.7838	0.3765	0.06165	0.4913	7.355	2.969	1.074e+0	0.972	0.7918	0.6715	0.8797
9	0.9938	0.5623	0.1219	0.1004	0.4235	8.802	0.09667	1.029e+0	0.03386	0.7543	0.5443	0.8849
10	0.9038	0.9271	0.2256	0.066	0.1612	1.648	9.363	1.032e+0	0.343	0.2897	0.3912	0.6542
11	0.9677	0.8308	0.05784	0.09437	0.1903	2.548	8.673	1.073e+0	0.7907	0.2467	0.6065	0.8313
12	0.9936	0.8488	0.3066	0.07129	0.6931	8.837	0.8767	1.066e+0	0.4323	0.7692	0.5471	0.932
13	0.9939	0.9899	0.1334	0.02434	0.4849	6.879	0.5613	1.051e+0	0.8321	0.6527	0.7817	0.6703
14	0.9938	0.5887	0.1008	0.06911	0.9129	3.438	1.432	1.355e+0	0.6516	0.5847	0.4622	0.6287
15	0.9481	0.7873	0.1396	0.1881	0.431	6.123	11.63	1.355e+0	0.2801	0.3822	0.5926	0.8202
16	0.9924	0.6325	0.4847	0.07778	1.014	5.0	0.1511	1.339e+0	0.4494	0.704	0.5079	0.8154
17	0.9631	0.5583	0.07293	0.1437	1.686	3.826	8.109	1.335e+0	0.1578	0.2312	0.6644	0.6346
18	0.989	0.9691	0.3055	0.1284	0.1529	7.437	1.284	1.348e+0	0.8166	0.3424	0.9346	0.9699
19	0.9822	0.5259	0.007432	0.1916	1.955	1.172	0.5653	1.365e+0	0.3355	0.5198	0.906	0.6424
20	0.9931	0.8673	0.1451	0.1811	1.357	9.077	0.2902	1.082e+0	0.4969	0.7158	0.4031	0.9113
21	0.9938	0.7079	0.1328	0.1104	0.2152	9.779	1.936	1.093e+0	0.803	0.5786	0.5353	0.6637
22	0.9547	0.5563	0.2574	0.1214	1.531	8.13	11.31	1.09e+03	0.1042	0.3746	0.2445	0.8372
23	0.9945	0.8537	0.3479	0.03114	1.882	9.626	0.3135	1.101e+0	0.8364	0.4653	0.429	0.6978
24	0.6867	1.0	0.09916	0.2	0.01	1.326	0.0	1.097e+0	1.0	1.0	1.0	0.7324
25	0.9906	0.6741	0.01225	0.05168	1.283	7.181	1.018	1.06e+03	0.692	0.3626	0.9354	0.778
26	0.9565	0.5392	0.169	0.1865	1.364	7.777	5.862	1.046e+0	0.5279	0.3108	0.2533	0.812
27	0.9757	0.7087	0.3096	0.1778	0.1826	1.737	3.897	1.055e+0	0.7436	0.2516	0.3485	0.9576
28	0.9871	0.9497	0.4117	0.05509	1.159	1.162	0.1698	1.349e+0	0.2317	0.3482	0.8045	0.8418
29	0.9914	0.7801	0.2487	0.05871	0.7787	9.258	0.3064	1.111e+0	0.3848	0.4308	0.8141	0.8453
30	0.9652	0.5	0.5	0.02	2.0	10.0	6.904	1.106e+0	0.3579	0.2	0.2	0.6

图 4.2 贝叶斯优化计算过程图

从图中可以看出，函数在第 23 次组合时便得到了算法认为的最优超参数组合，后续进行了 7 次重新组合均未取得更优的结果。在以上的 30 次构建的模型输出结果中，最高的 AUC 均值为 0.9945，输出的最优超参数组合取值为表 4.18 所示：

表 4.18 贝叶斯优化输出超参数表

超参数名称	取值
max_depth	10
learning_rate	0.031142588870517143
n_estimators	1101
min_child_weight	0
max_delta_step	1.8820901136533406
subsample	0.6978309406546113
colsample_bytree	0.8536734112346028
reg_alpha	0.8363848990156912
reg_lambda	0.46533848215278834
scale_pos_weight	0.42903081121182957

续表 4.18

gamma	0.3478500363393831

综合以上三次超参数自动调优的结果，贝叶斯优化结果相比其他两种算法有一定优势，所以最终本文选取了贝叶斯优化的调参结果来重新构建 XGBoost 模型。然后，利用训练好的模型对于测试集数据进行预测，得到预测结果的 AUC 值为 0.999207，ROC 曲线如图 4.3 所示：

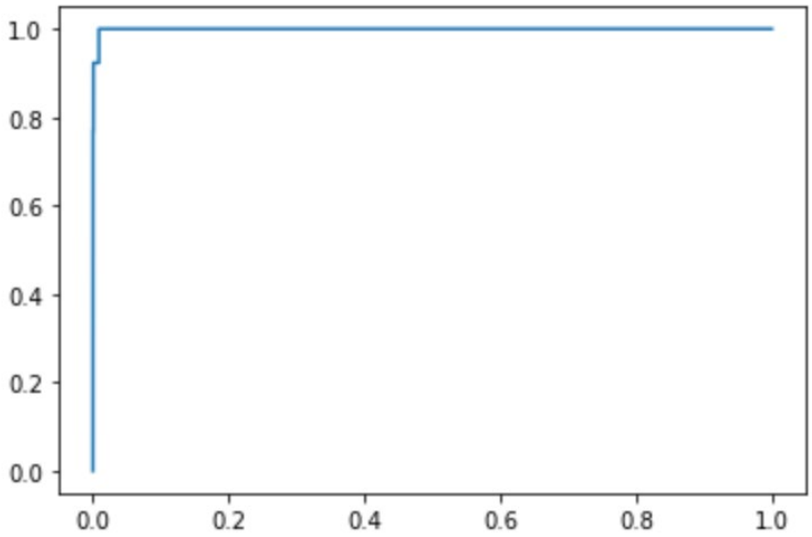

图 4.3 测试集 ROC 曲线

其次，将预测输出结果和实际结果导入到一张数据表中，通过计算得到本次预测结果的准确率为 99.928%，查准率（Precision）为$Precision = \frac{TP}{TP+FP} = 76.92\%$，查全率（Recall）为$Recall = \frac{TP}{TP+FN} = 62.5\%$，预测结果较为准确。最后，将这张数据表输出为 CSV 文件，文件命名为 ab.csv。通过观察得到，在测试集中，共有 13 只实际违约债券，模型成功预测出其中 10 只违约债券；模型输出的预测结果中有 16 只债券违约，其中有 6 只债券为误判。通过以上数据可以看出，模型具有一定的实用性，其在成功找出大部分实际违约债券的同时，并未将过多的实际未违约债券误判为违约债券。混淆矩阵如表 4.19 所示：

表 4.19 测试集混淆矩阵

Confusion Matrix		Actual class	
		Positive	Negative
Predicted class	Positive	TP=10	FP=6
	Negative	FN=3	TN=12511

4.4 模型迁移应用

以上过程表明，XGBoost 算法模型在国企、央企违约债券预测分类中具有良好的效果。接下来将利用此次训练好的模型，使用迁移学习的思想，对于城投债数据进行分析预测。

迁移学习的本质是将从某个领域中学习到的知识或是数据模型应用到相关领域中以解决实际问题。在迁移学习中，将原有用于学习的数据集称为源域，将需要进行分析学习的数据集称为目标域。

按照特征空间性质分类，迁移学习可以分为同构迁移学习，即源域与目标域有相同的特征空间；和异构迁移学习，即两者的特征空间不同。异构迁移学习相比同构迁移学习更为复杂，需要考虑的方面也比较多。本文在最初挑选特征变量时，便考虑到特征空间的一致性因素，所以本文使用的方法属于是同构迁移学习。

本文的数据集情况为源域中数据充足且均有标记，目标域数据同样充足但无标记，两个域的特征空间和标记空间均相同，但在数据数值大小和分布上有一定的差距。并且由于本文的主要任务为利用源域对于目标域进行分类预测，所以是属于直推式迁移学习。它是基于特征迁移的方式，即通过特征变化，使源域和目标域的特征投影在相同的空间，具有相同的分布，从而迁移使用模型。所以，在导入城投债数据集后，利用 StandardScaler

函数对其进行拟合转换，使得每列特征指标的数值均值为零，方差为一，即符合正态分布。最后，利用此前的主成分分析法得到的系数矩阵对于城投债数据集的特征变量进行降维。

在将目标源的特征处理结束后，便可以使用此前训练好的 XGBoost 模型对城投债数据进行分类预测，将预测结果和预测分类结果的概率各自作为新的列插入数据表的末尾，分别命名为‘pred’和‘pred_proba’。为了提高观察数据的便利性，提取数据表的‘id’、‘pred’、‘pred_proba’三列数据，输出为 CSV 文件，将其命名为‘result.csv’。通过观察得到，其中共有数据 13964 行，不存在模型认为有违约概率的债券，即模型认为 2020 年中城投债不会发生违约，这一结果符合实际情况，在一定程度上证明了模型的有效性。

同时，因为 XGBClassifier 分类器本质预测的并不是准确的 0 或 1 的分类，而是预测其属于某一分类的概率。于是在得到的预测结果中，本次实验也获取了每一行债券的具体分类概率数据。基于上文中所述，2020 年的预测结果中不存在模型分类为有违约概率的债券，因此本文通过研究分类概率对于城投债的预警结果进行进一步的分析。由于整体数据量较多，下表 4.20 展示了降序后前 6 只债券分类结果的概率：

表 4.20 部分城投债分类概率表

id	pred_proba
152052.SH	0.004153227
124849.SH	0.003491919
031490616.IB	0.002951268
1480305.IB	0.002125506
124684.SH	0.0021138
1480082.IB	0.00203998

从上表 4.20 中，可以看出分类概率值普遍较小。为了能够分析以上概率值的具体实际信息，重新调取了此前的测试集预测结果的相关数据。在降序后，选取分类概率边界线左右 20 只连续债券的有关数据，具体如下表 4.21 所示：

表 4.21 部分测试集分类概率表

id	pred_proba
011902431.IB	0.9084269
112016102.IB	0.9025796
011901110.IB	0.8042166
112016018.IB	0.80360764
111916346.IB	0.3924882
111821174.IB	0.21338025
111821057.IB	0.21338025
163608.SH	0.21338025
111921073.IB	0.034830052
111799756.IB	0.031208333
102001572.IB	0.020306941
112006236.IB	0.012157588
112021470.IB	0.01027377
111871337.IB	0.009787748
111977470.IB	0.006412186
112016147.IB	0.006339246
111882148.IB	0.004438297
111693601.IB	0.00442406
111973459.IB	0.002975999

在研究了表 4.20 和表 4.21 后发现，虽然城投债预测结果的分类概率最高仅为 0.04 多，但是根据测试集的分类概率表可以看出，不能够仅凭借分类概率的数值来判断债券的违约可能。分类概率仅为模型拟合数据后得到的数值，与传统意义的违约概率有较大区别，需要通过对比模型在测试集中的表现来判断分类概率的实用意义。

在测试集数据中，分类概率交界处的两个值分别为 0.80360764 和 0.3924882，两者的

差距远大于分界处外的其他分类概率间隔，出现这种现象可能的原因是违约债券数据相比于整体较小，类高度不平衡。同时可以观察到在测试集的 12530 条数据中，输出概率为 0.004 的债券与预测违约债券中的间隔仅有 12 只，可以看出与预测违约债券的实际差距非常小。所以，对于同样输出分类概率 0.04 左右的城投债也不能够放松警惕，需要提高重视。

综上所述，迁移后的模型预警结果有两方面的数据可以用于避免无序违约的发生。首先可以先观察是否存在有违约可能的债券，对于此类债券需要加强监控，及时进行干预；其次，在对于前一类债券采取措施之后，可以根据模型的分类概率结果对于排序靠前的债券提前进行风险排查，防患于未然。

第 5 章 结论与展望

5.1 研究结论

本文通过文献分析法、比较分析法、实证研究法等方法，结合与债券有关四个方面的特征指标（债券自身、发债主体、宏观经济、地区经济）和 XGBoost 算法，对于债券是否违约这二分类任务进行学习研究。具体步骤为：首先利用国企、央企债券数据来训练模型，其次对于模型的有效性进行优化并输出结果，然后基于迁移学习的思想，对城投债数据进行预测。本次测试中，得到了以下结果：

（1）XGBoost 算法模型在实际比较中表现出明显优势，模型得到的 AUC 值大于其余七种机器学习模型：随机森林、ET、高斯朴素贝叶斯、K 最邻近、逻辑回归、决策树、支持向量机。这说明 XGBoost 算法适用于债券违约风险预测。

（2）在三种机器学习领域主流的三种超参数优化算法中，贝叶斯优化和其余两种调参算法相比，使模型最终 AUC 值提高了 1.9‰。这说明贝叶斯优化算法对于模型有一定的改善效果。在原本 AUC 值较高的情况下，进一步地优化了模型的分类预测能力。

（3）模型在测试集预测中，AUC 值为 99.92%，查准率（Precision）为$Precision = \frac{TP}{TP+FP} = 76.92\%$，查全率（Recall）为 $Recall = \frac{TP}{TP+FN} = 62.5\%$，F1 值为 $F1 = 2\frac{Precision*Recall}{Precision+Recall} = 68.96\%$。这说明模型能够有效地学习数据分类出违约债券。

（4）本文基于迁移学习的思想，利用训练好的模型得到了城投债数据的预测结果。结果表明 2020 年不存在模型认为的违约债券，这符合 2020 年城投债违约的实际情况。说明本文构建的风险预警模型在城投债违约预测有实用意义。

本文通过创新性的信用风险预警模型，在机器学习方法于城投债违约风险探究的实际应用方面进行了拓展。

5.2 未来展望

本文虽取得了初步的研究结果，但仍存在着不足之处和有待提升之处，主要有以下几点可以作为进一步研究的方向：

（1）城投债的发债主体地方融资平台虽然是国有企业的一部分，但其特殊的金融、财政属性，使得它与国企、央企债券还是存在一定的差距。在进行迁移学习时，模型的分类预测效果不免有损失，后续可以通过域自适应算法（Domain Adaptation）对两者的数据集进行域对抗训练，以减少迁移学习过程中的不适配程度。

（2）可改用 Stacking 框架对于模型进行优化：结合多种机器学习算法模型的优势以提高整体模型分类结果的准确性。同时，从特征工程角度，可以根据特征指标的重要性结果对特征指标进行筛选，将每种机器学习算法得到的相对不重要指标列出，对于那些在多数情况下都被选取的特征指标进行删除以避免干扰模型，降低效果。

（3）在特征指标的增添优化方面，目前债券的发债主体所处地区均使用省级数据，但在实际中，国家级企业、省级企业、市级企业和县级企业具有各自特点和不同的性质。本文由于公开数据的不完整性，使得本次数据选取只能停留在省一级。后续如果能够获取相关详细数据，可以将地区指标进一步细化到县，从而匹配县级的地区经济指标以进一步提高模型分类的效果。

5.2 未来趋势和建议

依据国家政策走向，未来城投债的刚性兑付很有可能被打破。为了避免无序违约的乱象发生，债券市场监管部门需要时刻警惕，及时利用一定手段区别出潜在违约债券并进行风险排查，及时对市场投资者进行风险预警。根据历史和其他国家的做法，债务问题通常通过以下的三种方法进行控制：开源节流、财政赤字货币化、破产重组。

本文的建议是在破产重组方面，市场监管部门可以实时利用自身的信息优势，基于本文思路，构建更为准确有效违约风险预警模型，以区别不同质量的城投债。对于那些存在风险的城投债及时调整评级。同时根据模型输出结果，对于失去清偿能力的地方融

资平台及时督促其进行破产重组，以避免发生无序违约事件使市场不稳定。

参考文献

[1] 寇宗来,盘宇章,刘学悦.中国的信用评级真的影响发债成本吗?[J].金融研究,2015, 10:81-98.

[2] McInish, Thomas H . THE DETERMINANTS OF MUNICIPAL BOND RISK PREMIUMS BY MATURITY[J]. Journal of Financial Research, 1980, 3(2):129–138.

[3] Poterba J M , Rueben K S . State Fiscal Institutions and the U.S. Municipal Bond Market[J]. NBER Working Papers, 1997, 33(2):181-208.

[4] Gupton G M, Finger C C, Bhatia M. CreditMetrics-Technical Document, JP Morgan & Co[J]. Incorporated, 1997.

[5] Bharath S T , Tyler S . Forecasting Default with the KMV-Merton Model[J]. SSRN Electronic Journal, 2004, 21.

[6] 涂盈盈. 城投债的发展与风险控制[J]. 中国金融, 2010, 000(007):45-47.

[7] 何杨,满燕云.地方政府债务融资的风险控制——基于土地财政视角的分析[J].财贸经济,2012, 05:45-50.

[8] 罗荣华,刘劲劲.地方政府的隐性担保真的有效吗?——基于城投债发行定价的检验[J].金融研究,2016, 04:83-98.

[9] 钟辉勇,钟宁桦,朱小能.城投债的担保可信吗?——来自债券评级和发行定价的证据[J].金融研究,2016, 04: 66-82.

[10] 王未卿,肖勇贵,李霞.基于随机森林回归模型的城投债信用利差影响因素研究[J].数学的实践与认识,2020, 50(12):311-320.

[11] 肖叶,刘小兵.财政支出偏向是否促进了土地出让规模扩张[J/OL].当代经济科学,2021, 1-14.

[12] 刘红忠,许友传.地方政府融资平台债务重构及其风险缓释[J].复旦学报(社会科学版),2017, 59(06):143-154.

[13] 朱莹,王健.市场约束能够降低地方债风险溢价吗?——来自城投债市场的证据[J].金

融研究,2018, 06:56-72.

[14] 韩文丽,谭明鹏.监管趋严背景下地方政府融资平台债务现状、评判及对策探析[J].西南金融,2019, 01:55-63.

[15] 徐军伟,毛捷,管星华.地方政府隐性债务再认识——基于融资平台公司的精准界定和金融势能的视角[J].管理世界,2020, 36(09):37-59.

[16] 罗玉,肖丽雯,杨泽鹏.基于机器学习方法的城投债偿债风险评估[J].对外经贸,2019, 10:102-104.

[17] 杨雨茜. 复杂网络、融资平台杠杆与城投债信用风险传导[D].厦门大学,2019.

[18] 徐继伟,杨云.集成学习方法:研究综述[J].云南大学学报(自然科学版),2018, 40(06):1082-1092.

[19] Dasarathy B V, Sheela B V. A composite classifier system design: Concepts and methodology[J]. Proceedings of the IEEE, 1979, 67(5): 708-713.

[20] Hansen L K, Salamon P. Neural network ensembles[J]. IEEE transactions on pattern analysis and machine intelligence, 1990, 12(10): 993-1001.

[21] Freund Y, Schapire R E. A decision-theoretic generalization of on-line learning and an application to boosting[J]. Journal of computer and system sciences, 1997, 55(1): 119-139.

[22] Jacobs R A, Jordan M I, Nowlan S J, et al. Adaptive mixtures of local experts[J]. Neural computation, 1991, 3(1): 79-87.

[23] Freund Y, Schapire R E. A decision-theoretic generalization of on-line learning and an application to boosting[J]. Journal of computer and system sciences, 1997, 55(1): 119-139.

[24] Friedman J, Hastie T, Tibshirani R. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)[J]. Annals of statistics, 2000, 28(2): 337-407.

[25] 方敏. 集成学习的多分类器动态融合方法研究[J]. 系统工程与电子技术, 2006, 11:1759-1761.

[26] 何鸣, 李国正, 袁捷,等. 基于主成份分析的Bagging集成学习方法[J]. 上海大学学报

(自然科学版), 2006, 12(4):415-418.

[27] 李国正, 李丹. 集成学习中特征选择技术[J]. 上海大学学报:自然科学版, 2007, 05:598-604.

[28] 付忠良. 通用集成学习算法的构造[J]. 计算机研究与发展, 2013, 04:861-872.

[29] 刘伍颖, 王挺. 一种多过滤器集成学习垃圾邮件过滤方法[C]. 全国信息检索与内容安全学术会议, 2007.

[30] 谢倩倩, 李订芳, 章文. 基于集成学习的离子通道药物靶点预测[C]. 湖北省计算机学会学术年会. 湖北省计算机学会, 2014.

[31] 孙菲菲,林平,曹卓.基于旋转森林集成学习的涉恐实体挖掘研究[J].情报杂志,2015, 34(05):190-195.

[32] 张燕平, 窦蓉蓉, 赵姝,等. 基于集成学习的规范化LDA人脸识别[J]. 计算机工程, 2010, 14:144-146.

[33] 唐小果, 李毓. 个人信用评估应用方法分析——基于集成学习算法视角[J]. 经济问题, 2011, 12:66-68.

[34] 刘俊生,赵倩.地方政府债务风险评估及可持续性研究——基于KMV模型对苏浙两省的分析[J].北方金融,2020, 10:38-42.

[35] 杨剑锋,乔佩蕊,李永梅,王宁.机器学习分类问题及算法研究综述[J].统计与决策,2019, 35(06):36-40.

[36] 刘丽艳,朱成全.机器学习在经济学中的应用研究[J].天津师范大学学报(社会科学版),2020, 02:51-58.

[37] Chen T，He T，Benesty M，et al. Xgboost: extreme gradient boosting[J]. R package version 0.4-2，2015, 1-4.

[38] 费凡,谢灏,朱姝娟.城投债信用利差影响因素分析[J].债券,2020, 07:70-74.

[39] 徐兆丰,杨妹,肖淇棱.城投债信用风险的影响因素分析——以广东省债券为例[J].时代金融,2020, 31:80-82+91.

[40] 岳庆生. 基于不平衡数据的XGBoost性能优化研究[D]. 兰州交通大学, 2019.

附 录

1 主成分分析法转换矩阵

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	0.008916	-0.26815	-0.31977	-0.04534	0.017789	0.003469	-0.00443	0.002012	-0.00534	-0.0041	0.006244	-0.00248	-0.00157	-0.00026	-0.01021	0.004152	0.00598	0.100015
1	-0.26051	-0.01362	0.012389	-0.08626	-0.01152	0.038717	0.158748	0.176742	-0.15844	0.2054	-0.15251	0.037004	-0.21922	0.036897	0.413965	-0.06395	0.008825	-0.25962
2	0.201556	0.027363	-0.0231	-0.05377	-0.17084	0.132666	0.452707	-0.01366	0.018561	0.048566	-0.02339	-0.12606	0.160591	-0.26502	0.140299	0.243317	-0.25793	0.148942
3	0.141826	0.002712	0.002822	0.000344	0.232265	0.070915	0.468531	0.140386	0.310674	-0.07302	-0.08998	-0.127	0.103444	-0.33539	0.055201	0.154687	0.276435	0.02917
4	-0.17911	-0.04433	0.026168	-0.07103	0.113964	-0.32119	0.151375	-0.05873	0.102311	-0.13164	-0.2195	0.555344	0.091523	-0.28714	-0.02745	0.102435	-0.09785	-0.15987
5	-0.02009	-0.0153	-0.00675	-0.05621	-0.35159	-0.12826	0.382393	-0.29391	0.003853	0.042502	-0.13912	-0.0108	0.101385	0.597379	-0.00636	0.241882	0.386818	-0.01318
6	0.01393	0.01263	0.007042	0.013103	0.361777	0.246067	0.212798	0.022511	0.086528	-0.44141	-0.34231	-0.26267	-0.20914	0.359666	-0.1004	-0.11872	-0.34982	-0.10637
7	0.003398	-0.01894	-0.00972	0.00042	-0.30969	-0.15089	0.001746	0.419015	0.428194	-0.28152	0.125039	-0.07967	-0.39851	-0.04985	-0.01448	-0.21428	0.331293	0.037521
8	0.036584	0.02207	0.010919	0.036218	0.485441	0.272349	0.097604	-0.10789	-0.10169	0.136334	0.034955	0.133214	0.096912	-0.03123	0.030258	-0.32243	0.549912	-0.00189
9	0.068382	-0.01244	0.024857	0.052634	0.171815	0.226969	-0.35971	0.482193	0.047381	0.026096	-0.18234	0.105571	0.130099	0.195903	0.076957	0.624014	0.148825	0.045305
10	0.129128	0.010305	-0.00619	0.011012	0.067387	0.18237	0.25914	0.201508	0.297067	0.242024	0.457623	0.499117	-0.03327	0.354278	-0.06864	-0.08804	-0.28188	-0.00847
11	-0.07899	0.163937	-0.11813	-0.03785	0.260273	0.045221	0.018331	-0.31247	-0.11536	-0.08252	0.339305	-0.00125	-0.44624	-0.00949	-0.03337	0.284936	0.075179	0.227891
12	0.018076	-0.31496	0.259032	0.055614	0.025568	0.000961	0.018014	-0.06596	-0.01687	-0.02087	0.077483	-0.00867	-0.10676	-0.02647	-0.0095	0.091132	0.003884	-0.026
13	0.233942	-0.01653	0.006752	0.025746	-0.15072	0.245824	0.046663	0.123113	-0.15507	0.272131	0.233719	-0.32629	0.031687	-0.09838	0.047159	-0.04339	-0.01423	-0.26282
14	0.008951	-0.2684	-0.32013	-0.04539	0.017794	0.003539	-0.00447	0.001907	-0.00542	-0.00396	0.006138	-0.00241	-0.00134	-0.00018	-0.00957	0.004028	0.005621	0.091072
15	0.018544	-0.31752	0.257262	0.055626	0.024804	0.001343	0.016516	-0.06108	-0.01607	-0.01938	0.07056	-0.00781	-0.09637	-0.02334	-0.00838	0.081108	0.003405	-0.01783
16	0.00901	-0.26836	-0.31985	-0.04523	0.017053	0.00311	-0.00445	0.001766	-0.00494	-0.003	0.005967	-0.00229	-0.00127	-0.00065	-0.00953	0.004237	0.006578	0.101611
17	0.018607	-0.3147	0.261316	0.056456	0.024247	0.001774	0.015596	-0.05888	-0.01574	-0.01883	0.069184	-0.00775	-0.09479	-0.02327	-0.00819	0.082024	0.002566	-0.02988
18	-0.0071	0.212742	0.262091	0.039143	-0.02417	-0.0148	0.006854	0.005023	0.01312	0.007313	-0.00303	0.001327	-0.01968	-0.01267	-0.03216	0.006754	0.030015	0.618705
19	-0.02136	0.248854	-0.18582	-0.04471	0.004443	-0.01085	0.029795	-0.09806	-0.01076	-0.04172	0.19641	-0.01473	-0.33805	-0.15337	-0.05734	0.395916	0.018704	-0.34377
20	0.008978	-0.26848	-0.32019	-0.04537	0.017612	0.003456	-0.00448	0.001837	-0.00533	-0.00368	0.006061	-0.00235	-0.00124	-0.00027	-0.00935	0.004039	0.005739	0.090705
21	0.018542	-0.31383	0.262808	0.056529	0.024441	0.001768	0.015917	-0.05906	-0.01587	-0.01864	0.067862	-0.00754	-0.09215	-0.02205	-0.00747	0.077825	0.002683	-0.02606
22	0.00911	-0.26752	-0.3185	-0.04482	0.015341	0.002419	-0.00454	0.001069	-0.0038	-0.0005	0.005528	-0.00207	-5.58E-05	-0.00144	-0.00798	0.004488	0.008043	0.106811
23	0.019005	-0.3149	0.260214	0.05709	0.021935	0.002322	0.012019	-0.05051	-0.01387	-0.01604	0.060395	-0.00793	-0.0813	-0.01676	-0.00697	0.071191	0.000647	-0.0251
24	-0.06548	0.012753	-0.07915	0.42486	-0.07837	0.094859	0.038094	-0.02683	0.004435	0.021058	-0.04005	0.012555	-0.04067	-0.02185	0.113093	-0.01858	-0.01382	-0.05217
25	-0.15965	0.009713	0.004004	0.053408	-0.04999	0.029258	0.142077	0.222985	-0.36898	-0.49931	0.379057	0.010247	0.373132	0.013114	-0.30267	0.028613	0.104486	-0.12944
26	-0.30582	-0.01458	-0.01909	0.127202	0.063288	-0.01777	0.014873	-0.00277	0.140229	0.089036	0.117774	-0.18464	0.079037	0.015025	0.082755	0.028236	-0.10786	0.258165
27	-0.1903	-0.00986	-0.02039	0.161285	-0.02296	-0.05388	0.243718	0.287368	-0.29525	0.274224	-0.16846	-0.00844	-0.17017	-0.07461	-0.47692	0.014359	-0.0688	0.179691
28	-0.15785	-0.02789	0.021722	-0.10471	-0.2407	0.489196	-0.08402	-0.16571	0.142846	0.045394	-0.18855	0.14334	-0.12164	-0.1287	-0.24978	-0.01187	0.103064	-0.035
29	0.092567	0.018947	-0.07554	0.408606	-0.09883	0.105065	-0.03559	-0.11618	0.071079	-0.01247	-0.0949	0.090579	-0.02164	-0.05158	-0.13713	0.006392	0.020272	-0.06203
30	-0.04717	0.009811	-0.0746	0.422045	-0.04252	0.054526	-0.03332	-0.12103	0.165486	0.04856	-0.05802	0.009071	0.033186	-0.03089	-0.14646	0.027108	0.021193	-0.10698
31	0.297594	0.028853	-0.04404	0.184178	-0.13319	0.125349	-0.04223	-0.0824	-0.08104	-0.15185	-0.05789	0.143149	-0.03896	-0.04742	0.021401	-0.02688	-0.01822	0.096099
32	0.122223	0.025028	-0.05908	0.337219	0.146453	-0.29654	-0.00397	-0.02551	0.026375	-0.06547	0.120286	-0.05569	0.05814	0.078436	0.328958	0.007245	-0.03088	0.027612
33	-0.29824	-0.01005	-0.036	0.237472	0.015116	0.02489	0.050411	0.011657	0.10275	0.120225	0.02309	-0.09996	0.043643	-0.01718	-0.01822	0.043459	-0.04531	0.024342
34	0.349635	0.026931	-0.02371	0.053677	-0.04724	-0.01049	-0.06166	-0.09421	0.00321	-0.08425	-0.04505	0.080548	0.020116	-0.01602	-0.0692	-0.01199	-0.01957	-0.00935
35	-0.17221	0.002335	-0.06409	0.372338	-0.04811	0.048896	0.044529	0.005958	-0.05877	-0.11511	0.028239	0.080976	-0.00205	0.012092	0.196429	-0.00138	0.045918	-0.04651
36	0.354474	0.024971	-0.01419	0.021553	0.012419	-0.08262	-0.0606	-0.0717	0.03367	0.002516	-0.05321	0.030012	0.002022	-0.01957	-0.15833	0.001637	-0.01924	-0.01032
37	-0.1927	-0.0146	0.013664	-0.09926	-0.22963	0.413663	-0.02486	-0.09719	-0.01185	-0.29883	0.082072	0.158541	0.093597	-0.08819	0.341166	-0.01468	-0.08149	0.161784
38	-0.22374	-0.00394	0.022173	-0.08912	0.092874	0.002783	-0.14121	-0.19609	0.467415	0.075182	0.158476	-0.24856	0.314362	-0.00185	-0.19915	0.062144	-0.0499	-0.14657

2 原始特征指标重要性表

原始特征指标重要性表

特征指标名称	重要性值
breachNewYear	0.290492
breachNewYearGDP	0.165069
breachNewYearGDPgrowthindex	0.059685
bondIssuePrice	0.057451
regCap	0.024492
bondBalance	0.043829

续上表

bondIssuerRating	0.023508
sbreachNewYearMinimumLivingSecurity	0.023349
sbreachNewYearUnemploymentRate	0.020253
region	0.020009
bondTerm	0.019367
industry	0.018742
sbreachNewYearPowerUsage	0.017546
sbreachNewYearGDPGrowthIndex	0.017259
sbreachNewYearGPBE	0.017121
sSSFR	0.013855
sbreachNewYearHousePrice	0.013814
sbreachNewYearPNGT	0.011863
comScale	0.011802
bondInterest	0.011694
sbreachNewYearPD	0.010955
sbreachNewYearGDP	0.010930
typeBond	0.009745
sbreachNewYearGPBR	0.009120
guaCom	0.008970
sbreachNewYearCPI	0.008847
sbreachNewYearPCDI	0.008626
bondTotalAmt	0.008568
intBeginDate	0.008330
sbreachNewYearUrbanArea	0.007541
intBeginDateGDP	0.006531

续上表

sbreachNewYearCityNumber	0.006900
intBeginDateGDPgrowthindex	0.006457
intBeginDateFinanceScaleRate	0.006382
intBeginDateConsumptionLevel	0.000897
breachNewYearConsumptionLevelIndex	0.000000
breachNewYearConsumptionLevel	0.000000
breachNewYearFinanceScaleRate	0.000000
intBeginDateConsumptionLevelIndex	0.000000

致 谢

在这里，我想衷心地感谢我的指导老师在论文写作过程中给予我的帮助和建议。导师精益求精的治学态度和对专业知识的独到见解使我获益良多。起初时，我难以找到论文中所需的相关数据，老师的帮助使我最后成功找到了数据的来源，开始论文的第一步。

同时感谢在写作论文时，在构思和方法上给予我有力帮助的文章作者。没有前人在相关领域的研究和创作文章时的无私奉献，本次毕业论文便难以完成。在实际论文写作中，我遇到了不少起初未曾想到的困难，正是他们的帮助使我逐步理清思路，确定方法以及最终所用的方法。

其次感谢我的舍友在百忙之中每日不忘提醒我及时写论文，并时常在宿舍中研讨论文中遇到的问题。感谢他们时常拉我去打羽毛球，使我在这几月中没有放弃锻炼，也使我有更长的时间构思创作论文。

我谨向所有关心和帮助过我的老师、同学和家人致以真诚的谢意。

褚天舒

2021 年 5 月 15 日于东大浑南校区