基于随机森林回归模型的文旅现象对地方经济影响问题研究

摘要

本文针对“尔滨”现象对哈尔滨市经济影响的问题，通过建立随机森林回归模型，实现了文旅现象对哈尔滨经济的影响的求解。

经过综合考虑并结合互联网数据，我们选取：旅游总收入（亿元）、接待旅游者人数（万人次）、百度指数“哈尔滨”搜索量、哈尔滨市城镇居民人均年消费支出（元）、省外游客占比%、商品零售价格指数、失业率%、环境质量：可吸入颗粒物年均值（毫克/立方米）作为主要特征因素进行数据分析。数据预处理阶段，我们利用 linear 线性插值的方式对数据空缺值进行填充，以保证数据完整性。再经过标准化处理，以消除不同特征之间的量纲影响，使得不同特征在模型中能够更加公平地比较和权衡。我们将旅游总收入（亿元）作为目标变量，其余因素作为特征变量。按照 7：3 比例划分为训练集和测试集，进入随机森林回归模型中进行训练和预测。经过相关系数计算，得到对目标变量影响最大的是城镇居民人均年消费支出，相关系数为 0.984563。最后对该市未来 5 年（2024-2028）的旅游总收入进行了预测，并进行了数据可视化。在评估方面我们对该套模型进行均方根误差 (RMSE)、平均绝对误差 (MAE)、R^2 参数的评估，分别得到 146.01、103.08、0.9086 的成绩。可见模型性能良好。

通过本文的研究，我们深入探讨了旅游文化现象对哈尔滨经济的影响，并建立了完整的随机森林回归模型来进行预测和分析。我们的模型展现出了较高的准确性和可靠性，在评估文旅现象对经济的影响方面提供了有效的解决方案。随机森林回归模型在处理复杂数据和预测问题上具有出色的性能，为我们提供了一个强大的工具，帮助我们更好地理解和预测文旅现象对地方经济的影响。

关键词：随机森林回归　机器学习　相关系数　文旅现象

一、问题重述

伴随着互联网的发展，新兴的文化旅游现象已经成为推动地方经济发展的重要力量。近年来，地方文化和旅游现象频频“出圈”，吸引了大量的关注和讨论，成为推动地方经济发展的重要引擎。哈尔滨，作为中国东北的一座重要城市，以其独特的冰雪文化和丰富的旅游资源，吸引了众多国内外游客。本文的目标是定量评估哈尔滨文旅现象对地方经济的影响，为当地文旅局提供科学的决策依据。

二、问题分析

2.1 数据量

一个地方的旅游业发展影响因素有很多，诸如接待旅游者人数、网站搜索量、旅游基础设施投资，甚至在宏观层面失业率、就业人数等方面都会对该地旅游业发展具有一定影响。但结合当地政府官方统计局已有的数据，我们没办法完全分析每一个具体的因素对方经济发展的影响。故为保证模型的准确性，本文只选取了官方数据较多的八大因素：旅游总收入（亿元）、接待旅游者人数（万人次）、百度指数“哈尔滨”搜索量、哈尔滨市城镇居民人均年消费支出（元）、省外游客占比%、商品零售价格指数、失业率%、环境质量：可吸入颗粒物年均值（毫克/立方米）进行数据分析。

2.2 数据选择

在年份选择方面上，由于 2020-2022 年三年的疫情因素，大大影响了全国各地区的旅游业发展。考虑到这三年数据的大幅度下跌会对模型预测值有重大影响以及目前国内旅游业趋于常态化。故为确保模型预测值的真实性，本文将 2020-2022 年的数据不纳入分析范围内。

2.3 分析目标

问题要求定量评估文旅现象对地方经济的影响力，其问题可以转化为通过特征重要性来理解各个特征对地方旅游业经济的贡献度。转换成数学语言即，计算出各个特征变量与目标变量的相关系数，其值介于-1 到 1 之间。正数说明该因素对地方旅游业经济起促进作用，负数说明该因素对地方旅游业经济起抑制作用。

2.4 模型的选择

基于此类问题我们可以选择以下几种模型：

- 线性回归模型

  在简单线性关系下，线性回归模型的参数估计具有较好的效果，且计算速度快，可解释性强。但在旅游经济中，往往存在复杂的非线性关系，线性回归模型可能无法很好地捕捉到这些关系，导致预测精度下降。

- 时间序列分析模型（ARIMA）

  ARIMA 模型能够捕捉到时间序列数据的趋势和周期性，对于探索旅游经济的季节性变化具有一定的优势。然而，ARIMA 模型对于非线性关系的处理能力有限，可能忽略了与其他因素的复杂交互作用，影响了对旅游经济的全面理解。

- Logistic 回归模型

  其适用于二分类问题，可以估计概率，有助于预测旅游需求的概率。但 Logistic 回归模型假设数据线性可分，可能无法捕捉到复杂的非线性关系，限制了其在旅游经济预测中的应用范围。

- 主成分分析（PCA）

  主成分分析能够降维并提取关键特征，有助于简化模型和减少过拟合。然而，PCA 可能会损失信息，对非线性数据不够灵活，可能无法完全解释旅游经济的复杂性。

- 支持向量机（SVM）

  在高维空间中表现良好，有助于处理复杂的非线性关系，提高了模型的预测精度。但 SVM 对参数敏感，且计算复杂度较高，可能不够适用于大规模数据集的建模。

- 随机森林回归（RFR）

  随机森林能够有效处理高维度的数据和大规模的样本，适用于旅游经济中可能存在的大量影响因素。相比一些简单的线性模型，随机森林在训练过程中需要消耗更多的计算资源和时间，特别是在处理大规模数据集时可能会面临挑战。

相比于其他模型，随机森林对数据的预处理要求较少，减少了建模前的数据处理时间和复杂度。其能够灵活地处理非线性关系和复杂的交互效应，提高了模型的预测能力。同时也能够提供特征重要性评估，有助于深入理解旅游经济中各因素的影响程度，为决策提供重要参考。综合考虑，本文选择随机森林回归模型作为最终模型。

2.5 模型搭建环境

主流的数学建模环境有 Matlab 和 Python。Matlab 是一种专注于数学计算和工程领域的高级编程语言和交互式环境。它具有丰富的数学函数库和绘图功能，适合进行数值计算、仿真、数据分析以及算法开发等任务。Matlab 的语法简洁清晰，对于处理矩阵和向量等数学对象十分方便，是许多科学工程领域的标准工具之一。Python 是一种通用编程语言，可读性好，易于维护和分享，有助于团队协作和项目管理。且具有丰富的第三方库和生态系统，拥有大量成熟的第三方库，特别是在科学计算和数据分析领域。例如，NumPy 提供了高效的多维数组操作，SciPy 提供了科学计算函数和工具，pandas 提供了灵活强大的数据结构和数据分析工具，scikit-learn 提供了机器学习算法和工具等，这些库的组合使得在 Python 中进行数学建模更加高效。综合考虑本文选用 Python 作为模型搭建环境。

三、模型假设

- 假设样本数据是独立同分布的，即每个样本之间的观测结果相互独立。
- 假设特征之间存在一定的相关性，但没有严重的多重共线性问题。故忽略其影响因素，以确保模型的稳定性和可靠性。
- 假设模型中的树之间是相互独立的，每棵树的构建不受其他树的影响，以确保集成学习的效果。
- 认为附件所有数据真实可靠。
- 忽略季节天气、交通状况等不确定影响。
- 疫情因素可能引发市民的消费习惯变化，从而影响城镇居民人均年消费支出等指标。为考虑模型复杂度，故忽略该影响因素。

四、符号说明

符号	说明
RMSE	均方根误差
MAE	平均绝对误差
R^2	拟合优度

表 4-1 符号说明

五、模型的建立与求解过程

5.1 数据预处理

5.1.1 Linear 线性插值

在原始的数据中（原始数据见附录一），由于商品零售价格指数、失业率%、环境质量：可吸入颗粒物年均值（毫克/立方米）部分年份无官方数据。考虑到不同年份间可以通过已知数据点之间的线性关系来推断缺失值，故本文选用 Linear 线性插值法来填充缺失值。代码实现中使用了 Python 第三方库 Pandas 中的 interpolate() 方法，对 DataFrame 中的数值列进行线性插值填充缺失值。将填充后的完整数据写入新文件 preprocessed_data.xlsx 中。

5.1.2 变量划分

创建两个新列表，将接待旅游者人数（万人次）、百度指数“哈尔滨”搜索量、哈尔滨市城镇居民人均年消费支出（元）、省外游客占比%、商品零售价格指数、失业率%、环境质量：可吸入颗粒物年均值（毫克/立方米）七大因素划归为列表 X，作为特征变量。将旅游总收入（亿元）划归为列表 Y，作为目标变量。

5.1.3 标准化特征

考虑到不同特征的取值范围和单位可能不同，这会导致模型受到特征尺度的影响而表现不稳定。而标准化可以将所有特征转换为具有相同尺度的值，消除了量纲带来的影响，使得不同特征之间可比较性更强。同时标准化也可以加快优化算法的收敛速度，提高训练模型的效率。利用 scikit-learn 库中的 sklearn.preprocessing 模块，实例化了一个标准化器 StandardScaler，然后使用该标准化器对特征数据集 X（特征变量）

进行标准化处理，即将特征数据缩放到均值为 0，标准差为 1 的范围内。

5.1.4 数据切割

分割比例的选择是经验性的，并没有一个固定的标准。我们在保证其他因素不变的情况下，分别用常用的几种切割比例进行测试，得到下列 R^2 的评分（越靠近 1 模型效果越好）。详见表 5-1 数据切割测试。数据得出结论，按照 7：3 的切割方式效果最好。故我们将 30%的数据分配给测试集，而剩下的 70%则分配给训练集。

train：test	R^2
8:2	0.8421
7:3	0.9086
6:4	0.8975
5:5	0.7228

表 5-1 数据切割测试

5.2 模型建立

5.2.1 随机森林回归原理

随机森林回归是一种基于集成学习的算法，它通过将多个决策树的预测结果进行平均或加权平均，从而得到最终的回归结果。其核心思想是利用多个决策树模型，获得比单个决策树更高的预测精度。单个决策树的性能并不一定高，但是多个决策树汇总起来，就能够创建出泛化能力更强的模型。图解随机森林模型见图 5-1。其解题步骤如下所示：

- Step1：随机选择样本集

从训练数据集中随机选择一个样本集，这个样本集的大小通常和原始训练数据集的大小相同，但是每个样本的选择是随机的，并且可能会有重复。

- Step 2：随机选择特征

对于每个决策树的训练过程中，从所有特征中随机选择一个子集。这个子集的大小通常小于总特征数，这样可以保证每个决策树的差异性。

- Step 3：训练决策树

使用步骤 1 和步骤 2 中选择的样本集和特征子集，训练一个决策树模型。通常使用基

尼系数或信息增益等指标来进行节点的划分，直到达到停止条件（如树的深度达到预定值）为止。

- Step 4：重复步骤 2 和步骤 3

重复多次步骤 2 和步骤 3，生成多棵决策树。每棵树都是通过不同的样本集和特征子集训练得到的，因此它们之间具有一定的差异性。

- Step 5：预测

对于回归问题，随机森林通过对每棵树的预测结果进行平均，得到最终的预测结果。

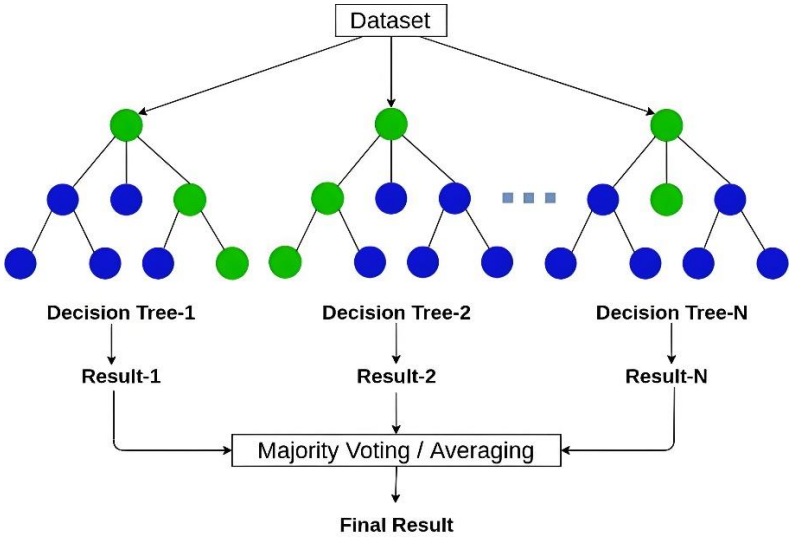

图 5-1 随机森林

5.2.2 模型创建与参数

在 Scikit-learn 库中，可以使用 RandomForestRegressor 类来构建随机森林回归模型，可以设置参数来控制随机森林的行为。一下为具体参数：

- n_estimators：决策树的数量

  通常情况下，增加决策树的数量可以提高模型的性能，但也会增加计算复杂度。一般来说，选择一个合适的数量，使得模型在性能和计算复杂度之间取得平衡。经过反复测试和尝试，选择树数量为 80 效果最佳。

- max_depth：决策树的最大深度

  控制决策树的生长深度，避免过拟合。较小的深度可能导致模型欠拟合，而较大的深度可能导致模型过拟合。经过反复测试和尝试，选择最大深度为 20 效果最佳。

- min_samples_split：节点分裂的最小样本数

  控制决策树节点分裂的最小样本数。如果某个节点的样本数少于该值，则不再进行分裂。可以通过设置较大的值来防止过拟合。

- random_state：随机种子

  设置随机种子可以使模型的随机性可复现，便于调试和比较不同模型的性能。本文设置了随机种子为 6，确保模型的随机性可复现。

5.3 模型评估

5.3.1 预测效果参数

本文采用 RMSE、MAE、R^2 三种参数进行模型性能评估，一下为具体概述：

- RMSE（均方根误差）：为 MSE 的平方根，取值越小，模型准确度越高。
- MAE（平均绝对误差）： 绝对误差的平均值，能反映预测值误差的实际情况。取值越小，模型准确度越高。
- R^2（拟合优度）： 将预测值跟只使用均值的情况下相比，结果越靠近 1 模型准确度越高。

参数	*RMSE*	*MAE*	R^2
测试集	*146.01*	*103.08*	*0.9086*

表 5-2 预测效果参数

指标显示该模型在测试集上表现良好，RMSE 和 MAE 较小，R^2 接近 1，说明模型的预测与真实值之间的拟合程度较高。因此，可以认为该模型在回归任务上具有较好的性能，能够满足预期。

5.3.2 模型拟合效果

为了更好地展示模型的性能效果，我们充分利用 matplotlib 的便携性，对拟合效果进行数据可视化。我们将随机抽样出的测试集样本与之对应的预测集进行对比比较，画出对比曲线图。如图 5-2 所示。可见模型预测值与真实值的曲线十分贴合，直观地展现出我们的优良的模型性能。绘图代码详见附录四。

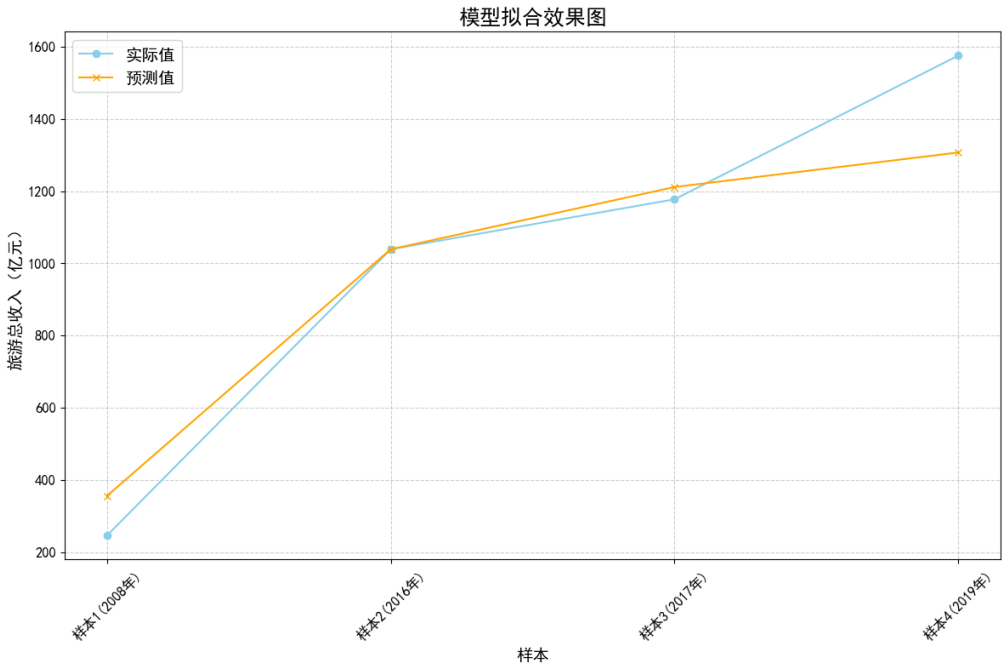

图 5-2 模型拟合效果图

5.4 相关性计算

在相关性计算方面，我们利用了 Pandas 库中 DataFrame 对象的 loc 方法。该方法允许通过标签或布尔数组访问一组行和列，来获取特征变量与目标变量（旅游总收入）之间的相关性。这样做可以快速而准确地计算出每个特征与目标变量之间的关联程度，从而帮助我们了解特征对于旅游总收入的影响程度。每个特征变量与目标变量的相关系数如下表 5-3 所示。

特征变量名	相关系数[-1,1]
接待旅游者人数（万人次）	0.981098
百度指数“哈尔滨”搜索量	0.687933
城镇居民人均年消费支出（元）	0.984563
省外游客占比%	0.849504
商品零售价格指数	-0.095914
失业率%	0.602428
环境质量：可吸入颗粒物年均值（毫克/立方米）	0.674708

表 5-3 相关系数

为更直观地呈现出数据，我们运用 matplotlib 对数据进行可视化处理，并对相关系数最高的特征变量进行标亮处理。如图 5-3 所示。

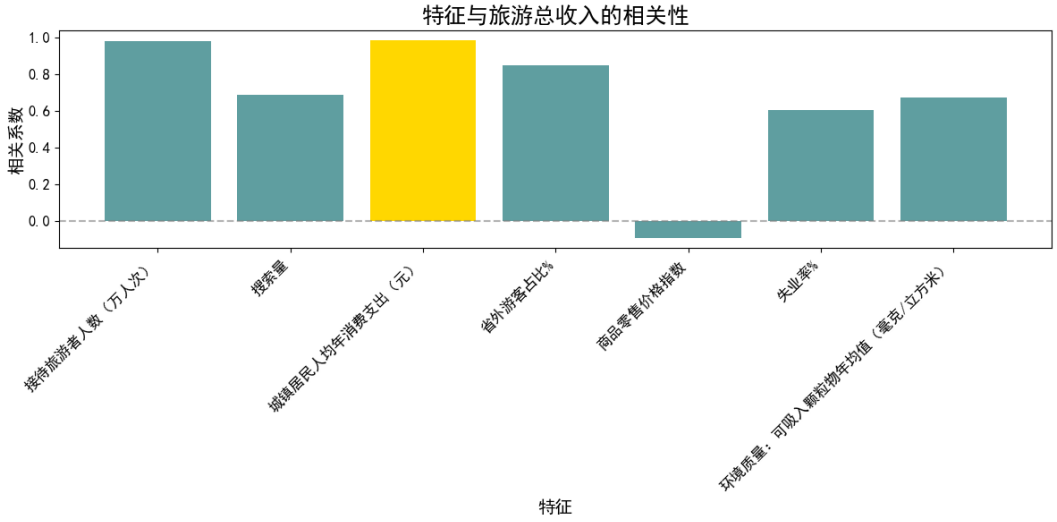

图 5-3 特征与旅游总收入的相关性

经过我们的模型分析，发现哈尔滨旅游总收入与哈尔滨城镇居民人均年消费支出（元）以及哈尔滨接待旅游者人数（万人次）之间存在着较高的相关性。这意味着提高哈尔滨城镇居民的消费水平和增加旅游者人数可能会直接促进旅游总收入的增长。针对此发现，我们建议哈尔滨文旅局在制定旅游发展策略时重点关注以下几个方面：

首先，我们认为哈尔滨在提升城镇居民消费水平方面有着广阔的空间。通过培育消费文化，打造具有哈尔滨特色的文化旅游产品和服务，可以吸引更多的城镇居民参与旅游消费。例如，可以加大对哈尔滨本地特色小吃、手工艺品、民俗文化体验等方面的推广力度，让居民在享受美食、购物、文化体验的同时，提高对旅游产品的认知和参与度，从而促进旅游总收入的增长。

其次，我们建议哈尔滨进一步提升旅游接待能力和服务水平。除了加大对旅游基础设施的投资外，还应该注重提升服务质量和提升游客体验。培训更多的旅游从业人员，提高他们的专业素养和服务意识，同时利用科技手段提升旅游信息化水平，提供更便捷、更个性化的旅游服务，让游客在哈尔滨留下美好的回忆，愿意再次光顾，从而实现旅游总收入的增长。

另外，我们认为哈尔滨文旅局可以进一步挖掘文化与旅游产业的融合潜力。通过

举办文化艺术节、展览、演出等活动，将哈尔滨独特的历史文化和地域特色展示给游客，提升他们的文化体验和参与度，从而吸引更多的游客前来旅游，促进旅游总收入的增长。

最后，我们强调创新对于哈尔滨文旅业发展的重要性。除了开发具有地方特色的旅游产品外，还可以借助科技手段，打造虚拟旅游体验、智慧导游等新型旅游产品，吸引更多年轻人和科技爱好者来哈尔滨旅游，拓展旅游市场，增加旅游总收入。

5.5 未来预测

为了为当地文旅局提供更有价值的意见，我们充分利用模型的预测能力对哈尔滨市未来 5 年（2024-2028）的旅游总收入进行了预测。我们使用了已知数据中的最后 5 年数据（2016 年至 2023 年），通过数据标准化和随机森林回归模型进行预测，得到了未来 5 年的目标变量的预测值。这种方法的优势在于利用了最近的数据，更能反映当前市场情况和趋势变化，从而提高了预测的准确性。2008-2028 年的所有数据写入到 predict_data.xlsx 文件中，详见附录三。

通过采用 matplotlib 对预测结果进行可视化处理，我们可以更好地观察数据走势，为文旅局提供更直观的数据分析和决策支持。详见图 5-4 2008-2028 年哈尔滨市旅游总收入走势图。根据走势图，旅游总收入在 2024 年有可能会有一定幅度的下跌，但后续会逐渐上升。总体水平与现在趋于平衡态势。

鉴于走势图显示旅游总收入在疫情前呈上升趋势，文旅局应持续收集和分析旅游数据，以便及时调整策略，把握旅游业的复苏态势。

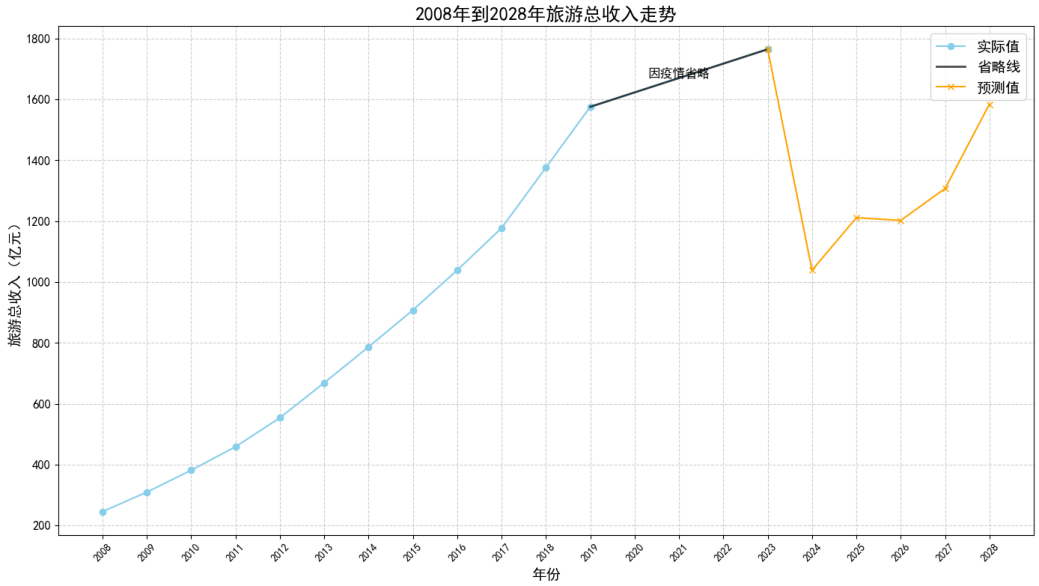

图 5-4 2008-2028 年哈尔滨市旅游总收入走势图

六、缺点与改进

6.1 数据预处理环节的缺点及改进方案

6.1.1 数据选择的主观性

缺点：由于 2020-2022 年的数据由于疫情影响被排除，这可能导致模型无法学习到疫情对旅游业的特定影响。

改进方案：考虑使用时间序列分析模型如 ARIMA 或 LSTM 神经网络来处理时间序列数据，这些模型可以捕捉到时间序列中的动态变化和趋势。**线性插值的局限性：**

6.1.2 线性插值的局限性

缺点：线性插值可能无法捕捉到数据的非线性特征，特别是在旅游业这样的季节性明显的行业中。

改进方案：考虑使用基于机器学习的插值方法，如基于随机森林或梯度提升树的回归模型，以预测缺失值，这些模型能够捕捉数据中的非线性关系。

6.1.3 特征选择的不足

缺点：仅考虑了八大因素，可能未涵盖所有影响旅游总收入的潜在变量。

改进方案：采用特征选择算法，如基于树模型的特征重要性评估，结合 LASSO 或弹性网等正则化技术，以识别和选择对预测目标影响最大的特征。

6.2 模型选择与评估指标的缺点及改进方案

6.2.1 模型解释性不足

缺点：随机森林模型作为一个黑盒模型，其决策过程不够透明。

改进方案：结合使用可解释的模型，如决策树或逻辑回归，以及模型解释工具，如 SHAP（SHapley Additive exPlanations）或 LIME（Local Interpretable Model-agnostic Explanations），来提供模型预测的可解释性。

6.2.2 评估指标的单一性

缺点：仅使用 RMSE、MAE 和 R^2可能无法全面评估模型的性能。

改进方案：引入额外的评估指标，如 Brier 分数、逻辑损失或 ROC-AUC，特别是对于分类问题或概率预测问题，以获得更全面的模型性能评估。

七、参考文献

[1]杨思祺,李淑兰.基于机器学习—线性回归算法的收入与用户预测模型在审计项目中的应用[J].无线互联科技,2023,20(23):94-98.

[2]周志华. 机器学习[M]. 清华大学出版社, 2016.

[3]张玉叶,李霞.基于 Pandas+Matplotlib 的数据分析及可视化[J].山东开放大学学报,2023,(03):75-78.

[4]孙宝存,赵九欣 主编,外经 外贸 旅游 地方旅游主要经济指标,孙宝存,赵九欣主编,河北经济统计年鉴,中国统计出版社,1993,467,年鉴.

[5]王水俤,数学 系统建模与数学模型,刘菊兰 主编,中国出版年鉴,中国出版年鉴社,1996,516,年鉴.

[6]春世增 主编,旅游 哈尔滨旅游资源首次普查,春世增 主编,中国民族年鉴,中央民族大学出版社,1997,335,年鉴.

附录

支撑材料 304 -------|-- .idea
|-- data -----|-- data.xlsx
| |-- predict_data.xlsx
| |-- preprocessed_data.xlsx
|
|-- image ----|-- 2008 年到 2028 年旅游总收入走势图.png
| |-- 模型拟合效果图.png
| |-- 特征与旅游总收入的相关性.png
| |-- Pycharm 代码运行截图.png
|
|-- main.py

一、 原始数据

文件：data.xlsx

年份	旅游总收入（亿元）	接待旅游者人数（万人次）	搜索量	城镇居民人均年消费支出（元）	省外游客占比%	商品零售价格指数	失业率%	环境质量：可吸入颗粒物年均值（毫克/立方米）
2008 年	**246**	**2993**	**3794**	**10791.2**	**11**	**101.6**	**3**	**0.102**
2009 年	**310**	**3752**	**4912**	**12358.1**	**12**	**101.5**	**3**	**0.101**
2010 年	**382**	**4127**	**6109**	**13939.5**	**10**	**100.1**	**3.4**	**0.101**
2011 年	**459**	**4361**	**7272**	**16232.7**	**14**	**104.4**	**3**	**0.099**
2012 年	**554**	**5055**	**9160**	**17614.6**	**12**	**102.5**	**3.4**	**0.094**
2013 年	**669**	**5529**	**8442**	**18614.6**	**16**	**101.2**	**3.6**	**119**
2014 年	**787**	**5993**	**8918**	**20331.8**	**15**	**101.5**	**3.72**	**111**
2015 年	**908**	**6499**	**11940**	**21638.5**	**18**	**100.2**	**3.88**	**103**
2016 年	**1039.1**	**7040**	**11173**	**22961.7**	**20**	**101.6**	**3.76**	**74**
2017 年	**1177.5**	**7688.9**	**13578**	**24340.1**	**25**	**99.7**	**3.68**	**87**
2018 年	**1376.2**	**8543.7**	**10527**	**25678.8**	**20**	**100.7**	**3.76**	
2019 年	**1575.7**	**9544.2**	**7672**	**27347.9**	**22**	**102.2**	**3.51**	
2023 年	**1764.8**	**12566.8**	**11678**	**30719.4**	**45**			

二、 标准化后数据

文件：preprocessed_data.xlsx

年份	旅游总收入（亿元）	接待旅游者人数（万人次）	搜索量	城镇居民人均年消费支出（元）	省外游客占比%	商品零售价格指数	失业率%	环境质量：可吸入颗粒物年均值

								（毫克/立方米）
2008	**246**	**2993**	**3794**	**10791.2**	**11**	**101.6**	**3**	**0.102**
2009	**310**	**3752**	**4912**	**12358.1**	**12**	**101.5**	**3**	**0.101**
2010	**382**	**4127**	**6109**	**13939.5**	**10**	**100.1**	**3.4**	**0.101**
2011	**459**	**4361**	**7272**	**16232.7**	**14**	**104.4**	**3**	**0.099**
2012	**554**	**5055**	**9160**	**17614.6**	**12**	**102.5**	**3.4**	**0.094**
2013	**669**	**5529**	**8442**	**18614.6**	**16**	**101.2**	**3.6**	**119**
2014	**787**	**5993**	**8918**	**20331.8**	**15**	**101.5**	**3.72**	**111**
2015	**908**	**6499**	**11940**	**21638.5**	**18**	**100.2**	**3.88**	**103**
2016	**1039.1**	**7040**	**11173**	**22961.7**	**20**	**101.6**	**3.76**	**74**
2017	**1177.5**	**7688.9**	**13578**	**24340.1**	**25**	**99.7**	**3.68**	**87**
2018	**1376.2**	**8543.7**	**10527**	**25678.8**	**20**	**100.7**	**3.76**	**87**
2019	**1575.7**	**9544.2**	**7672**	**27347.9**	**22**	**102.2**	**3.51**	**87**
2023	**1764.8**	**12566.8**	**11678**	**30719.4**	**45**	**102.2**	**3.51**	**87**

三、 预测值数据

文件：predict_data.xlsx

年份	旅游总收入（亿元）
2008	**246**
2009	**310**
2010	**382**
2011	**459**
2012	**554**
2013	**669**
2014	**787**
2015	**908**
2016	**1039.1**
2017	**1177.5**
2018	**1376.2**
2019	**1575.7**
2023	**1764.8**
2024	**1038.985**
2025	**1211.528**
2026	**1202.655**
2027	**1307.203**
2028	**1584.2**

四、 Python 代码

```
import pandas as pd
import numpy as np
```

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt

# 设置 matplotlib 字体和解决负号显示问题
plt.rcParams['font.sans-serif'] = ['SimHei']  # 设置字体为黑体，适用于中文显示
plt.rcParams['axes.unicode_minus'] = False  # 解决负号显示问题

# 读取 Excel 文件数据
df = pd.read_excel("D:/大学自学学习资料/数学建模/2024 比赛/Code/随机森林回归/data/data.xlsx")

# 删除多余的索引列
df = df.reset_index(drop=True)

# 将年份列中的中文字符“年”去掉，并转换为数值类型
df['年份'] = df['年份'].apply(lambda x: int(x[:-1]) if isinstance(x, str) else np.nan)

# 用线性插值方法填充缺失值
df['商品零售价格指数'] = df['商品零售价格指数'].interpolate(method='linear')
df['失业率%'] = df['失业率%'].interpolate(method='linear')
df['环境质量：可吸入颗粒物年均值（毫克/立方米）'] = df['环境质量：可吸入颗粒物年均值（毫克/立方米）'].interpolate(method='linear')

# 导出预处理后的数据到新的 Excel 文件
preprocessed_file_path = "D:/大学自学学习资料/数学建模/2024 比赛/Code/随机森林回归/data/preprocessed_data.xlsx"
df.to_excel(preprocessed_file_path, index=False)
print(f"预处理后的数据已保存至 {preprocessed_file_path}")

# 保留用于分析的特征和目标变量
features = [
    '接待旅游者人数（万人次）',
    '搜索量',
    '城镇居民人均年消费支出（元）',
    '省外游客占比%',
    '商品零售价格指数',
    '失业率%',
    '环境质量：可吸入颗粒物年均值（毫克/立方米）'
]

# 选择旅游总收入作为目标变量 y，其余作为特征 X
```

```
X = df[features]
y = df['旅游总收入（亿元）']

# 数据预处理 - 标准化特征
scaler = StandardScaler()  # 实例化标准化器
X_scaled = scaler.fit_transform(X)  # 标准化特征数据

# 将标准化后的数据转换为 DataFrame 并保留列名
X_scaled = pd.DataFrame(X_scaled, columns=features)

# 分割数据集
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42)  # 按 7:3 分割训练集和测试集

# 建立模型
rf = RandomForestRegressor(n_estimators=80, max_depth=20, bootstrap=True, random_state=6)  # 实例化随机森林回归模型
rf.fit(X_train, y_train)  # 训练模型

# 预测测试集结果
y_pred = rf.predict(X_test)  # 用模型预测测试集

# 未来 5 年预测
future_years = np.arange(2024, 2029)  # 定义未来 5 年的年份（2024-2028）

# 用回归模型对未来特征进行变化并预测
last_known_data = X.iloc[-5:]  # 获取已知数据中的最后 5 条记录
future_features_scaled = scaler.transform(last_known_data)  # 标准化未来特征
future_features_scaled = pd.DataFrame(future_features_scaled, columns=features)  # 保留特征名称
future_predictions = rf.predict(future_features_scaled)  # 预测未来特征对应的目标变量

# 将预测结果和未来 5 年的预测数据合并到 DataFrame 中
prediction_df = pd.DataFrame({
    '年份': np.append(df['年份'].values, future_years),  # 合并现有年份和未来年份
    '旅游总收入（亿元）': np.append(y.values, future_predictions.flatten())  # 合并现有旅游收入和未来预测值
})

# 导出预测的数据到新的 Excel 文件
predict_file_path = "D:/大学自学学习资料/数学建模/2024 比赛/Code/随机森林回归/data/predict_data.xlsx"
```

```
prediction_df.to_excel(predict_file_path, index=False)
print(f"预测的数据已保存至 {predict_file_path}")

# 模型评估
mse = mean_squared_error(y_test, y_pred)  # 计算均方误差
rmse = np.sqrt(mse)  # 计算均方根误差
mae = mean_absolute_error(y_test, y_pred)  # 计算平均绝对误差
r2 = r2_score(y_test, y_pred)  # 计算 R^2 得分

# 输出结果
print(f"均方根误差 (RMSE): {rmse:.2f}")
print(f"平均绝对误差 (MAE): {mae:.2f}")
print(f"R^2 得分: {r2:.4f}")

# 绘制模型拟合效果图
plt.figure(figsize=(12, 8))  # 设置图表大小
sorted_indices = np.argsort(df.loc[y_test.index, '年份'])  # 获取按年份排序的索引
x_labels = [f"样本{i+1}({year}年)" for i, year in enumerate(df.loc[y_test.index, '年份'].values[sorted_indices])]  # 创建 x 轴标签
x_ticks = np.arange(len(x_labels))  # 创建 x 轴刻度
plt.plot(x_ticks, y_test.values[sorted_indices], label='实际值', color='skyblue', marker='o')  # 绘制实际值，浅蓝色线
plt.plot(x_ticks, y_pred[sorted_indices], label='预测值', color='orange', marker='x')  # 绘制预测值，橙色线
plt.title('模型拟合效果图', fontsize=18)  # 设置图表标题
plt.xlabel('样本', fontsize=14)  # 设置 x 轴标签
plt.ylabel('旅游总收入（亿元）', fontsize=14)  # 设置 y 轴标签
plt.xticks(ticks=x_ticks, labels=x_labels, fontsize=12, rotation=45)  # 设置 x 轴刻度和标签
plt.yticks(fontsize=12)  # 设置 y 轴刻度
plt.legend(fontsize=14)  # 显示图例
plt.grid(True, linestyle='--', alpha=0.6)  # 设置网格线
plt.tight_layout()  # 调整图表布局
plt.show()  # 显示图表

# 绘制所有特征与旅游总收入的相关性
plt.figure(figsize=(12, 6))  # 设置图表大小
corr_matrix = df.corr()  # 计算数据的相关性矩阵
feature_names = features  # 特征名称列表
corr_with_target = corr_matrix.loc[feature_names, '旅游总收入（亿元）']  # 获取特征与目标变量的相关性
print("特征与旅游总收入的相关系数：")
print(corr_with_target)
max_corr = max(corr_with_target)  # 获取最大相关性值
```

```
    colors = ['gold' if v == max_corr else 'cadetblue' for v in corr_with_target]  # 设置条形图颜色，最大相关性为金色，其余为青色
    plt.bar(feature_names, corr_with_target, color=colors)  # 绘制条形图
    plt.title('特征与旅游总收入的相关性', fontsize=18)  # 设置图表标题
    plt.xlabel('特征', fontsize=14)  # 设置 x 轴标签
    plt.ylabel('相关系数', fontsize=14)  # 设置 y 轴标签
    plt.axhline(y=0, color='gray', linestyle='--', alpha=0.6)  # 添加水平线
    plt.xticks(rotation=45, ha='right', fontsize=12)  # 设置 x 轴刻度和标签
    plt.yticks(fontsize=12)  # 设置 y 轴刻度
    plt.tight_layout()  # 调整图表布局
    plt.show()  # 显示图表

    # 去除 2020 年到 2022 年的数据，确保这些年份不出现在实际数据点中
    missing_years = [2020, 2021, 2022]  # 省略的年份
    # 获取 2019 年和 2023 年的实际值
    value_2019 = df.loc[df['年份'] == 2019, '旅游总收入（亿元）'].values[0]  # 获取 2019 年的实际值
    value_2023 = df.loc[df['年份'] == 2023, '旅游总收入（亿元）'].values[0]  # 获取 2023 年的实际值
    # 绘制 2008 年到 2028 年的目标变量走势
    years = np.append(df['年份'].tolist(), future_years)  # 合并现有年份和未来年份
    revenues = np.append(df['旅游总收入（亿元）'].tolist(), future_predictions)  # 合并现有旅游收入和未来预测值
    # 去除 2020 年到 2022 年的数据点
    actual_years = [year for year in df['年份'] if year not in missing_years]  # 省略的年份实际年份列表
    actual_revenue = [revenue for year, revenue in zip(df['年份'], df['旅游总收入（亿元）']) if year not in missing_years]  # 省略年份的实际收入列表
    plt.figure(figsize=(14, 8))  # 设置图表大小
    # 绘制 2008 年至 2018 年及 2023 年的实际值
    actual_years_exclude_missing = [year for year in actual_years if year <= 2019 or year >= 2023]  # 省略年份实际年份列表，只包含 2019 年之前和 2023 年之后的年份
    actual_revenue_exclude_missing = [revenue for year, revenue in zip(actual_years, actual_revenue) if year <= 2019 or year >= 2023]  # 省略年份实际收入列表，只包含 2019 年之前和 2023 年之后的收入
    plt.plot(actual_years_exclude_missing, actual_revenue_exclude_missing, label='实际值', marker='o', color='skyblue')  # 绘制实际值，浅蓝色线
    # 用黑色实线连接 2019 年和 2023 年
    plt.plot([2019, 2023], [value_2019, value_2023], color='black', alpha=0.7, linewidth=2, label='省略线')  # 绘制省略线，黑色
    # 在 2023 年至未来几年用橙色实线绘制预测值
    future_years_with_2023 = np.insert(future_years, 0, 2023)  # 在未来年份数组的开头插入 2023 年
    future_predictions_with_2023 = np.insert(future_predictions, 0, value_2023)  # 在未
```

```
来预测值数组的开头插入 2023 年的实际值
    plt.plot(future_years_with_2023, future_predictions_with_2023, label='预测值', marker='x', color='orange')  # 绘制预测值，橙色线
    plt.annotate('因疫情省略', xy=(2021, (value_2019 + value_2023) / 2), fontsize=12, color='black', ha='center')  # 添加注释“因疫情省略”
    # 设置图表标题和轴标签
    plt.title('2008 年到 2028 年旅游总收入走势', fontsize=18)  # 设置图表标题
    plt.xlabel('年份', fontsize=14)  # 设置 x 轴标签
    plt.ylabel('旅游总收入（亿元）', fontsize=14)  # 设置 y 轴标签
    # 设置 x 轴刻度和标签
    plt.xticks(np.arange(2008, 2029, 1), rotation=45)  # 设置 x 轴刻度
    # 设置 y 轴刻度
    plt.yticks(fontsize=12)  # 设置 y 轴刻度
    # 显示图例
    plt.legend(fontsize=14)  # 设置图例
    # 设置网格线
    plt.grid(True, linestyle='--', alpha=0.6)  # 设置网格线
    # 调整图表布局
    plt.tight_layout()  # 调整图表布局
    # 显示图表
    plt.show()  # 显示图表
```