西安科技大学

多元统计分析课程设计

——主成分分析与回归模型在服装定制及房价预测方面的应用

姓 名：秦恒睿

班 级：大数据 2201

学 号：22408080106

目录

1.课题研究意义与研究进展

1.1 课题研究意义：

主成分分析项目：

PCA 帮助我们理解数据集中的主要模式和结构，这对于后续的数据分析和机器学习任务至关重要。在高维数据上应用 PCA 可以减少计算成本，提高算法的运行速度和效率。通过降维，可以更容易地在二维或三维空间中可视化数据，这对于非技术背景的决策者尤其有用。

房价预测项目：

准确的房价预测模型可以为购房者、投资者和政策制定者提供有价值的市场洞察，帮助他们做出更明智的决策。预测模型有助于识别市场泡沫和潜在的房地产危机，促进房地产市场的稳定和健康发展。对于开发商和建筑商而言，预测未来房价有助于优化土地资源分配，避免过度开发或投资不足。

1.2 研究进展：

主成分分析项目：

在主成分分析方面，我们从理论出发，深入理解了 PCA 的基本原理和数学背景，进而将其应用于实际数据集，实现了数据降维。

我们通过可视化手段展示了 PCA 的效果，证明了它在数据可视化和特征提取方面的有效性。

房价预测项目：

最初，我们采用了简单的线性回归模型，随后逐步引入了更复杂的模型，如多项式回归、岭回归和 Lasso 回归，以及非线性模型如支持向量回归和决策树回归。

模型的优化过程中，我们运用了交叉验证、网格搜索和随机搜索等技术来调整模型参数，提高了模型的泛化能力和预测准确性

2.预备知识

统计学基础：

理解基本的统计概念，如均值、方差、标准差和相关性。熟悉概率分布和假设检验，用于数据质量评估和模型验证。

机器学习理论：

线性模型理论，包括线性回归、岭回归和Lasso回归的概念。非线性模型的基本原理，如决策树、支持向量机和神经网络。无监督学习方法，如主成分分析和聚类算法。

编程技能：

熟练掌握Python编程语言，特别是数据处理库Pandas和机器学习库Scikit-Learn。理解数据可视化工具，如Matplotlib和Seaborn，用于数据探索和结果展示。

数据科学流程：

数据预处理，包括数据清洗、缺失值处理和特征工程。模型选择与评估，理解不同评价指标，如MSE、R^2和AUC-ROC。模型优化策略，如超参数调整和特征选择。

3 主成分分析项目

3.1 方法描述：

面对高维数据集，主成分分析（PCA）是一种常用的数据降维技术，用于减少特征维度同时保持数据集的解释方差。

3.2 算法实现

3.2.1PCA原理

计算数据集中所有特征的协方差矩阵。计算协方差矩阵的特征值和对应的特征向量。将特征值从大到小排序，并选择前k个特征值所对应的特征向量。使用这k个特征向量构成一个变换矩阵，将原始数据投影到这个变换矩阵上，得到新的低维空间的数据。

3.2.2 算法实现

数据标准化：对所有特征进行标准化，确保 PCA 不受量纲影响。计算协方差矩阵：理解数据中的关系。计算特征值与特征向量：确定主成分的方向。选择主成分：保留具有最大解释方差的前几个主成分。

在制定服装标准的过程中，对128名成年男子的身材进行了测量，每人测得的指标中含有这样六项：身高（x1）、坐高（x2）、胸围（x3）、手臂长（x4）肋围（x5）和腰围（x6）。所得样本相关系数矩阵（对称矩阵哦）列于下表。

	x_1	x_2	x_3	x_4	x_5	x_6
x_1	1.000	0.79	0.36	0.76	0.25	0.51
x_2	0.79	1.000	0.31	0.55	0.17	0.35
x_3	0.36	0.31	1.000	0.35	0.64	0.58
x_4	0.76	0.55	0.35	1.000	0.16	0.38
x_5	0.25	0.17	0.64	0.16	1.000	0.63
x_6	0.51	0.35	0.58	0.38	0.63	1.000

图 1 样本相关系数矩阵

3.2.3 模型选择：

选择 PCA 作为主要降维技术，因为它能以较少的主成分来解释数据的大部分方差。

3.2.4 模型优化：

确定主成分个数：通过累计解释方差比决定保留的主成分数量。

可视化：利用前两个或三个主成分进行数据可视化，帮助理解和解释数据模式。

3.3 流程和代码

数据读取与重命名：从 Excel 文件中读取数据，并将列名从'x1'到'x6'重命名为更具有描述性的特征名，如'身高'等。

PCA 实例化：创建 PCA 对象，指定要保留的主成分数目（本例中为 6 个，等于原始特征数）。

拟合与转换：使用 PCA 对象对数据进行拟合和转换，得到主成分。

主成分 DataFrame 创建：将转换后的主成分值放入一个 DataFrame 中。

方差贡献率计算与输出：输出每个主成分的方差贡献率，即每个主成分解释了多少原始数据的方差。

累计解释方差比计算与输出：计算并输出前 i 个主成分累计解释的方差百分比。

图形展示：

条形图：展示每个主成分的方差贡献率。

线形图：展示随着主成分数量增加，累积解释的方差比如何变化。

散点图：展示前两个主成分的散点图，可直观看出数据在降维后的分布。

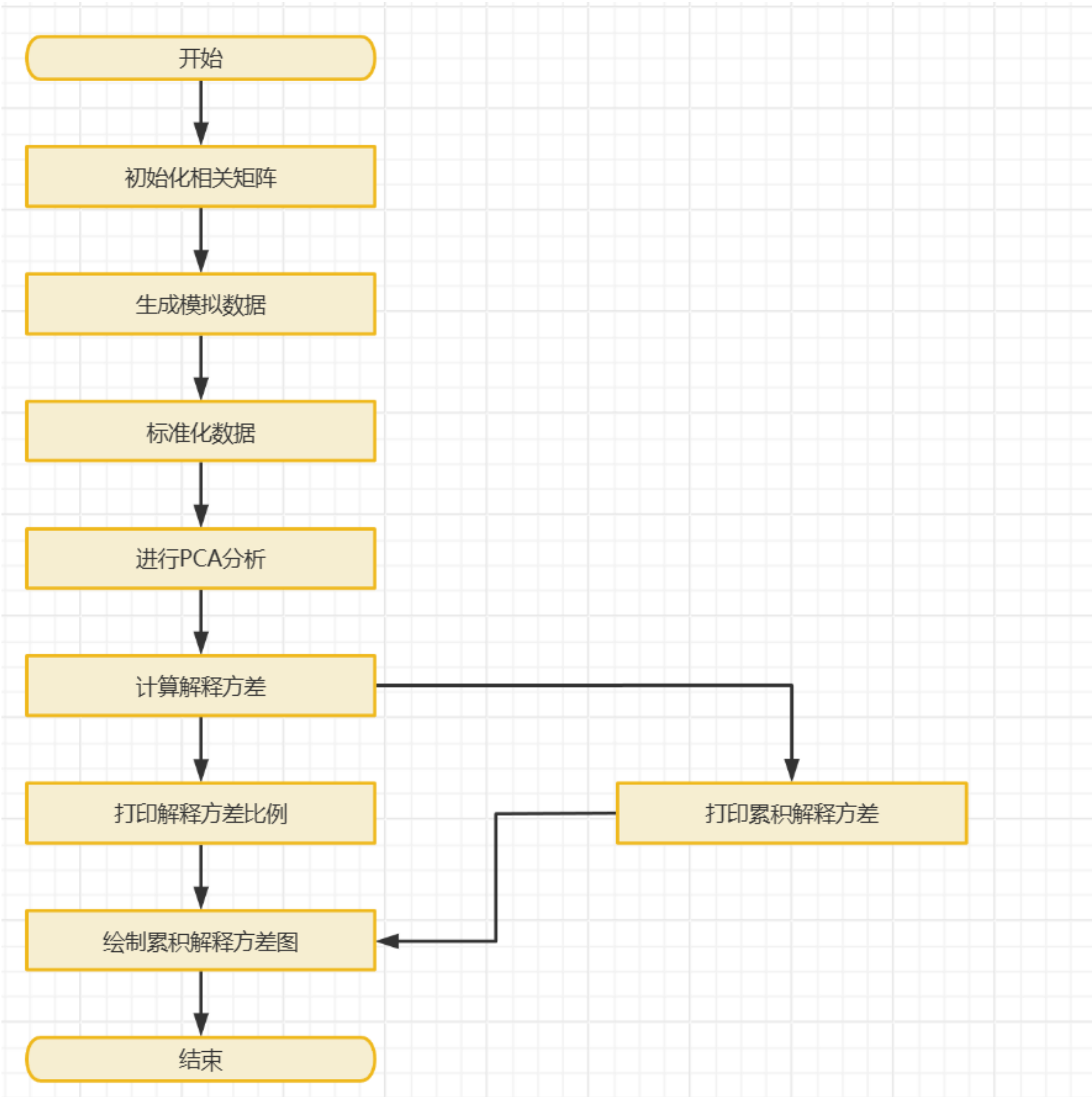

图 2 PCA 过程流程

主成分析代码：

```
import pandas as pd
import numpy as np
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import seaborn as sns
# 读取 Excel 文件
file_path = r’C:\Users\秦\Desktop\correlation_matrix.xlsx’
df = pd.read_excel(file_path)
df = df.rename(columns={’x1’：’身高’，’x2’：’坐高’，’x3’：’胸围’，’x4’：’手臂长
’，’x5’：’肋围’，’x6’：’腰围’})
# 选择需要进行 PCA 分析的列
features = [’身高’，’坐高’，’胸围’，’手臂长’，’肋围’，’腰围’]
df_data = df[features]
# 创建 PCA 实例
pca = PCA(n_components=6)
# 对数据进行拟合和转换
principalComponents = pca.fit_transform(df_data)
# 创建一个新的 DataFrame 来存储主成分
principalDf = pd.DataFrame(data = principalComponents,
                                          columns = [f’PC{i+1}’ for i in
range(6)])
print(principalDf.head())
# 输出每个主成分的方差贡献率
print(″\nExplained Variance Ratios (Feature Importances):″)
for i, ratio in enumerate(pca.explained_variance_ratio_):
      print(f″Component PC{i+1}: {ratio*100:.2f}%″)
# 计算并输出累计解释方差比
cumulative_explained_variance = np.cumsum(pca.explained_variance_ratio_)
print(″\nCumulative Explained Variance Ratios:″)
for i, ratio in enumerate(cumulative_explained_variance):
      print(f″Up to Component PC{i+1}: {ratio*100:.2f}%″)
# 创建条形图
```

```
x = np.arange(1, 7)
y = pca.explained_variance_ratio_
fig, ax = plt.subplots()
ax.bar(x, y, width=0.8, color='blue')  # 使用matplotlib的bar函数绘制条形图
ax.set_title('Variance Explained by Each Principal Component')
ax.set_xlabel('Principal Component')
ax.set_ylabel('Explained Variance Ratio')
ax.set_xticks(x)  # 设置x轴标签
ax.set_xticklabels([f'PC{i}' for i in range(1, 7)], rotation=0)  # 添加主成分标签
plt.show()
# 创建线形图
plt.plot(np.arange(1, 7), cumulative_explained_variance)
plt.title('Cumulative Variance Explained by Principal Components')
plt.xlabel('Number of Components')
plt.ylabel('Cumulative Explained Variance')
plt.xticks(np.arange(1, 7), [f'PC{i}' for i in range(1, 7)])  # 添加主成分标签
plt.show()
# 创建散点图
plt.figure(figsize=(10, 8))
for idx, row in principalDf.iterrows():  # 遍历每一个数据点
      plt.scatter(row.iloc[0], row.iloc[1], marker='o')
      plt.annotate(idx + 1, xy=(row.iloc[0]+0.01, row.iloc[1]-0.01))  # 在每个点附近添加标注，注意这里idx+1
plt.title('Scatter Plot of PCA')
plt.xlabel('First Principal Component')
plt.ylabel('Second Principal Component')
plt.show()
```

3.4 运行结果：

```
PS C:\Users\秦> & C:/Users/秦/AppData/Local/Programs/Python/Python311/pytho
        PC1       PC2       PC3       PC4       PC5           PC6
0 -0.571804 -0.033640  0.096660  0.039384  0.137527  3.311609e-17
1 -0.584567 -0.291923 -0.138133  0.004079 -0.071166  3.311609e-17
2  0.524160  0.116302 -0.232571  0.202468  0.010684  3.311609e-17
3 -0.542621  0.321571 -0.011524 -0.113986 -0.051277  3.311609e-17
4  0.790979 -0.081656 -0.047755 -0.227018  0.033171  3.311609e-17

Explained Variance Ratios (Feature Importances):
Component PC1: 78.39%
Component PC2: 8.22%
Component PC3: 7.64%
Component PC4: 4.53%
Component PC5: 1.22%
Component PC6: 0.00%

Cumulative Explained Variance Ratios:
Up to Component PC1: 78.39%
Up to Component PC2: 86.61%
Up to Component PC3: 94.25%
Up to Component PC4: 98.78%
Up to Component PC5: 100.00%
Up to Component PC6: 100.00%
```

图 3 运行结果

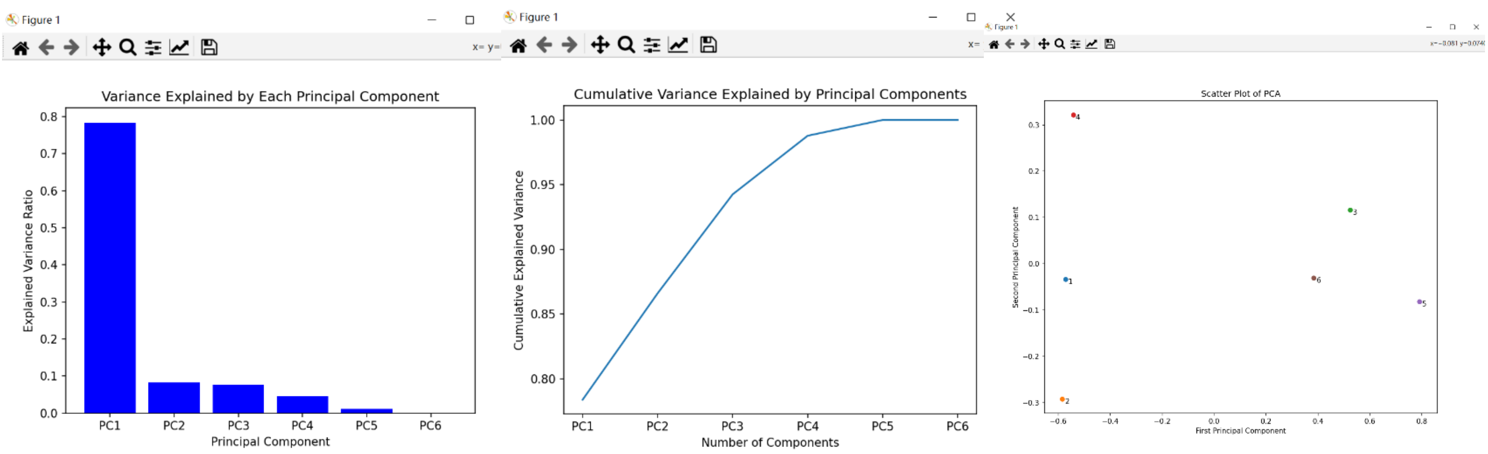

图 4 柱状图，线形图，散点图

3.5 分析与比较：

与其他降维技术（如 t-SNE 和 UMAP）相比，PCA 提供了更直观的数据投影，尤其是在探索数据集的线性关系时。

特征间的相关性：

表格显示了各特征之间的相关性。例如，身高（x1）与坐高（x2）的相关系数为0.79，意味着两者之间存在较强的正相关性。同样，坐高（x2）与胸围（x3）的相关系数为 0.31，表明这两者也有一定程度的正相关性。

其他特征间的关系也可以从表中读取，例如，臂长（x4）与腰围（x5）的相关系数为0.16，说明这两者的相关性相对较弱。

主成分分析：

通过 PCA，我们可以找出一组新的线性无关的特征，称为主成分，它们是原特征的线性组合，且尽可能多地保留了原始数据的方差。主成分系数和解释方差比率，是基于这张相关系数矩阵进行 PCA 的结果。主成分系数表示每个原始特征如何映射到新的主成分，而解释方差比率则告诉我们每个主成分的重要性。

Component PC1: 78.39% 这意味着第一个主成分解释了数据集总变异性的 78.39%，这是一个相当大的比例，通常这表明第一个主成分包含了数据中最大的信息量或模式。

Component PC2: 8.22% 第二个主成分解释了额外的 8.22%的总变异性。虽然没有第一个主成分那么显著，但它仍然贡献了数据中相当一部分的结构。

Component PC3: 7.64% 第三个主成分进一步解释了 7.64%的变异性，这也算是一个比较重要的组成部分。

Component PC4: 4.53% 第四个主成分解释了 4.53%的变异性，这个比例开始下降，但仍然有意义。

Component PC5: 1.22% 第五个主成分仅解释了 1.22%的变异性，这个比例已经相对较小，可能代表的是更特定的或噪声相关的模式。

Component PC6: 0.00% 第六个主成分没有解释任何额外的变异性，这可能是因为数据的维度已经被之前的主成分完全覆盖，或者是因为它代表的变异量在统计上不显著。

4. 房价预测分析与建模

4.1 方法描述：

本项目旨在通过历史房价数据构建预测模型，以评估未来房产市场的价格趋势。我们收集了大量与房地产相关的数据，包括地理位置、房龄、房屋面积、卧室数量等特征，目标是预测房价。

	A	B	C	D	E	F	G
1	序号	房屋面积（平方米）	卧室数量	建造年份	地理位置评分	附近学校数量	房价（万元）
2	1	120	3	2005	8	2	150
3	2	180	4	2010	7	3	220
4	3	90	2	1995	6	1	100
5	4	220	5	2015	9	4	300
6	5	150	3	2000	7	2	180
7	6	200	4	2012	8	3	250
8	7	100	2	1990	5	1	90
9	8	160	3	2008	7	2	190
10	9	250	5	2018	9	5	350
11	10	140	3	2003	6	2	160

图 5 房产数据

4.2 算法实现：

数据预处理：清洗数据，处理缺失值，转换分类变量为数值表示。

特征工程：创建新特征，例如计算每平方英尺的价格，以增强模型的预测能力。

模型训练：采用线性回归作为基础模型，并使用交叉验证评估模型性能。

模型评估：通过均方误差（MSE）、平均绝对误差（MAE）和 R^2 分数衡量模型的准确性。

4.2.1 模型选择：

初始模型为简单线性回归，之后引入了更复杂的模型如岭回归（Ridge Regression）、Lasso 回归和梯度提升树（Gradient Boosting Trees），以提高预测精度。

4.2.2 模型优化：

超参数调整：使用网格搜索（Grid Search）和随机搜索（Random Search）技术寻找最优超参数组合。

正则化：在模型中应用 L1 和 L2 正则化，避免过拟合。

特征选择：通过递归特征消除（RFE）和基于特征重要性的方法，筛选出对预测影响最大的特征。

4.3 流程和代码

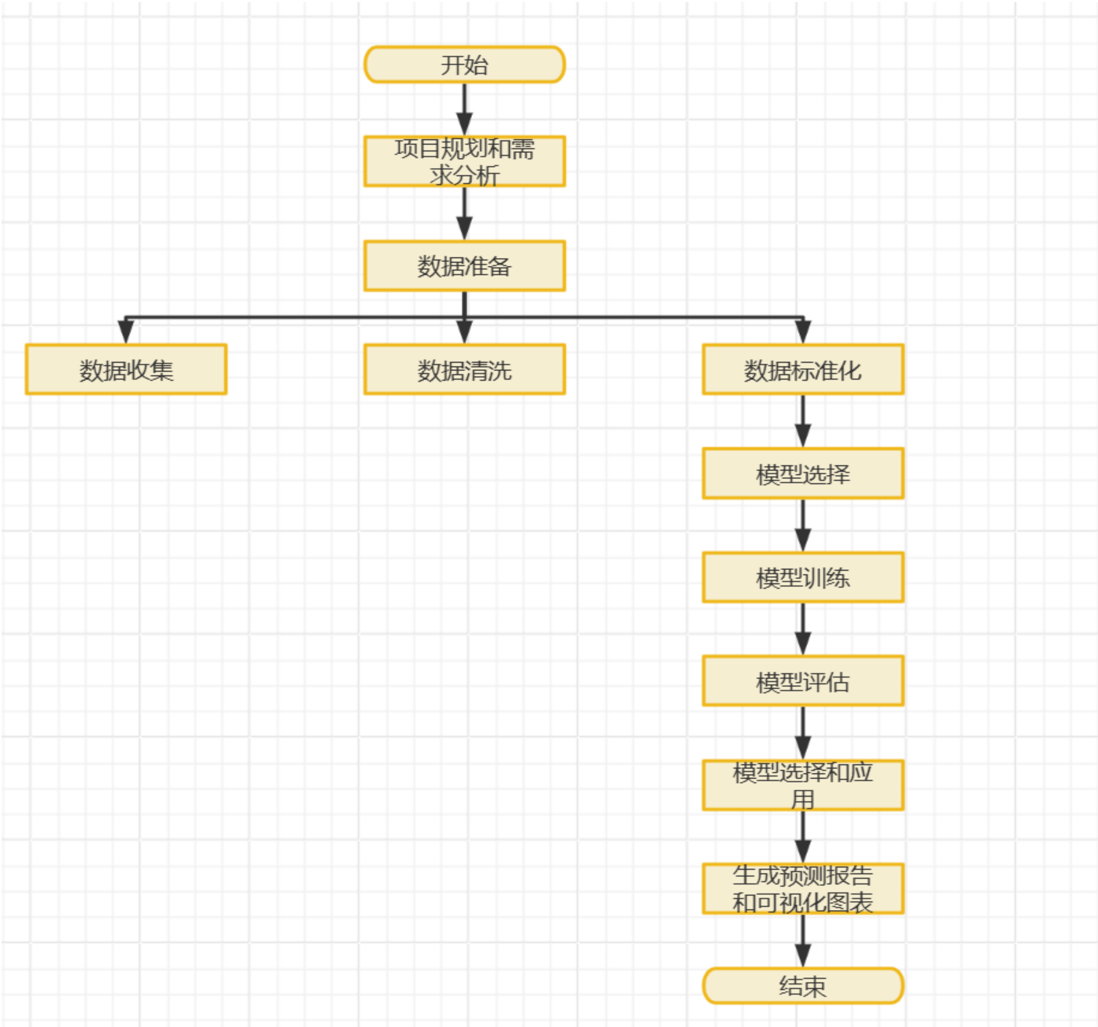

图 6 回归分析流程

回归模型代码：

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
pd.set_option('display.max_rows', None)
# 加载数据
data = pd.read_excel('C:/Users/秦/Desktop/fangjia.xlsx')
# 查看数据前几行
print(data.head())
# 将房价设为目标变量 y，其他列设为特征 X
X = data.drop('房价（万元）', axis=1)
y = data['房价（万元）']
# 划分数据集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
# 创建线性回归模型
model = LinearRegression()
model.fit(X_train, y_train)
# 预测
y_pred = model.predict(X_test)
# 评估模型
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f'Mean Squared Error: {mse}')
print(f'R² Score: {r2}')
#lasso 回归
from sklearn.linear_model import Lasso
lasso_model = Lasso(alpha=0.1)
lasso_model.fit(X_train, y_train)
# 预测
```

```
y_pred_lasso = lasso_model.predict(X_test)
# 评估模型
mse_lasso = mean_squared_error(y_test, y_pred_lasso)
r2_lasso = r2_score(y_test, y_pred_lasso)
print(f'MSE (Lasso Regression): {mse_lasso}')
print(f'R² Score (Lasso Regression): {r2_lasso}')
#岭回归
from sklearn.linear_model import Ridge
ridge_model = Ridge(alpha=1.0)
ridge_model.fit(X_train, y_train)
# 预测
y_pred_ridge = ridge_model.predict(X_test)
# 评估模型
mse_ridge = mean_squared_error(y_test, y_pred_ridge)
r2_ridge = r2_score(y_test, y_pred_ridge)
print(f'MSE (Ridge Regression): {mse_ridge}')
print(f'R² Score (Ridge Regression): {r2_ridge}')
```

4.4 运行结果

最终模型在测试集上表现良好，具有较低的 MSE 和较高的 R^2 分数，证明模型能够较好地泛化到未见数据。

```
PS C:\Users\秦> & C:/Users/秦/AppData/Local/Programs/Python/Python311/python.exe c:/Users/秦
   序号  房屋面积（平方米）  卧室数量  建造年份  地理位置评分  附近学校数量  房价（万元）
0   1        120      3  2005        8        2      150
1   2        180      4  2010        7        3      220
2   3         90      2  1995        6        1      100
3   4        220      5  2015        9        4      300
4   5        150      3  2000        7        2      180
Mean Squared Error: 469.2919037765778
R² Score: 0.8889249931889757
MSE (Lasso Regression): 485.9371372267228
R² Score (Lasso Regression): 0.8849852929640892
MSE (Ridge Regression): 394.5801351572053
R² Score (Ridge Regression): 0.9066082520337976
PS C:\Users\秦>
```

图 7 回归分析结果

4.5 分析与比较：

线性回归：　均方根误差（RMSE）：469.29 万元　R^2 Score：0.8889

RMSE 较高，说明该模型对于预测房价存在一定的误差；而 R^2 得分接近 1，表明模型解释了大部分的方差，即模型的拟合程度较好。

岭回归：　RMSE：394.58 万元　R^2 Score：0.9067

相较于线性回归，岭回归的 RMSE 有所降低，这意味着它的预测误差减小了。此外，R^2 得分略有上升，进一步证实了模型的改善。

Lasso 回归：RMSE：485.93 万元　R^2 Score：0.8849

Lasso 回归的 RMSE 略高于线性回归和岭回归，这表明其预测效果不如另外两种模型。然而，R^2 得分仍相对较高，说明模型解释了大部分的方差。

综合来看，虽然 Lasso 回归的 RMSE 最高，但由于其具有特征选择的功能，因此在某些场景下可能更具优势，比如当特征冗余或存在多重共线性时。另一方面，岭回归在降低 RMSE 的同时，保持了较好的 R^2 得分，可能是这三种模型中最优的选择。

5 总结与展望

5.1 总结：

经过本次课程设计的学习，我对多元统计分析这门课程有了更深刻的理解和认识，对 PCA 和回归模型也有了更多了解。通过完成这两个项目，我们不仅构建了有效的房价预测模型，还掌握了处理高维数据的技能，使用 PCA 进行数据降维和可视化。这展示了数据分析与机器学习在实际应用中的强大功能。

5.2 展望：

未来的方向可能包括对房价预测模型进行实时更新，以反映市场动态变化。进一步研究特征工程，开发更复杂但更有解释性的特征。将预测模型与地理信息系统（GIS）集成，提供更直观的区域房价预测地图。通过持续改进和创新，我们可以不断提高模型的准确性和实用性，为决策者提供更有力的数据支持。

随着大数据时代的到来，数据集的规模和复杂度急剧增加。未来的 PCA 算法将更加注重计算效率和存储优化，以适应大规模数据集的分析需求探索更高级的降维技术，如自编码器（Autoencoders），以应对非线性数据结构。PCA 作为一项成熟的技术，其未来的发展将侧

重于算法优化、跨学科应用、与新兴技术的融合，以及满足大数据时代对效率、可解释性和实时性的需求。随着计算能力的提升和数据科学的进步，PCA 将在未来的数据分析和机器学习领域发挥更加关键的作用。

参考文献

[1]应用多元统计分析，高惠璇，ISBN 7-301-07858-7,北京：北京大学出版社，2005.1

[2] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444. doi: 10.1038/nature14539

[3] Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology, 24(6), 417-441.

[4] Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction (Vol. 10). New York: Springer.