统计分析与建模期末项目

一、回归模型

1. 问题背景

问题背景介绍

- 本项目旨在通过构建多元线性回归模型，基于患者的多个特征变量，预测患者的满意度评分。随着医疗行业的不断发展，医院对于提升患者满意度的关注逐渐增加。患者的满意度不仅反映了医疗服务质量的高低，也直接影响着医院的声誉、患者复诊率以及医院的运营效率。因此，准确预测患者的满意度并采取针对性的干预措施，对于提升医院整体服务水平具有重要意义。
- 在本项目中，我们将使用以下自变量来预测患者的满意度：
  - **患者年龄（Age）**：反映患者的年龄段，年龄的不同可能影响患者的满意度。
  - **疾病严重程度指数（Severity）**：此数值表示患者的病情严重程度，病情越严重，患者的心理和生理负担越大，可能对满意度产生不同影响。
  - **是否为内科患者（Surg-Med）**：这是一个指示变量，区分患者是否为内科患者（0表示内科，1表示外科）。不同类型的治疗可能会影响患者的满意度。
  - **焦虑指数（Anxiety）**：此数值反映患者的焦虑程度，焦虑较高的患者可能对治疗过程和结果有更高的期待和更强的情绪反应，从而影响其整体满意度。
- **目标：**

  本项目的主要目标是利用多元回归模型分析上述自变量对患者满意度（Satisfaction）的影响，并通过模型评估患者满意度。通过准确的满意度预估，医院可以根据不同患者的情况，实施个性化干预措施，例如对潜在满意度评分较低的患者提供更及时的咨询服务，或者调整医疗服务流程，从而有效提升患者的整体满意度。
- **任务分解：**
  - **数据分析与描述统计**：首先，我们将通过描述统计学方法对数据进行全面分析，评估各个变量的基本特征，并为后续建模提供基础。
  - **数据建模与最优模型选择**：我们将基于多元线性回归模型，逐步筛选特征并进行模型训练，寻找最佳的回归模型。
  - **模型质量评估**：对建立的回归模型进行质量评估，包括检查模型的拟合度、残差分析等，确保模型的可靠性与准确性。
  - **影响因素分析**：通过对模型的回归系数进行解读，分析各个自变量对患者满意度的影响程度，为医院决策提供数据支持。
  - **模型优化与改进**：根据模型的表现，提出可能存在的问题，并提出改进方案，包括数据处理方法、特征选择优化等。
- 该项目不仅具有实际应用价值，帮助医院提高患者满意度，也为数据分析和建模提供了一个综合的实践机会，能够深入探讨多元回归分析在医疗领域的应用。

数据集概述

- 该数据集包含了25个患者的观察数据，其中包含以下列变量：
  - **Age (年龄)**：患者的年龄，影响患者健康状况及其对医疗服务的评价。
  - **Severity (疾病严重程度)**：疾病的严重程度指数，数值越高表示患者病情越严重，可能会影响患者对医院服务的满意度。
  - **Surg-Med (内科外科患者指示变量)**：一个二元变量，0表示内科患者，1表示外科患者。内科和外科患者的需求和治疗体验可能存在差异，进而影响满意度。
  - **Anxiety (焦虑指数)**：表示患者的焦虑程度，数值越高表示患者的焦虑越严重。焦虑可能会影响患者的情绪和对治疗的看法，从而影响满意度。

    - **Satisfaction (患者满意度)**：患者对医疗服务的整体满意度评分，作为目标变量，用于进行回归建模。
- 数据集中的每一行代表一个患者的不同特征及其对应的满意度评分。数据集共包含25个观察值，每个观察值对应着4个自变量和1个因变量。

2. 数据分析与描述统计

准备工作

```
# 加载必要的库
library(ggplot2)
library(dplyr)
library(tidyr)
library(gridExtra)

# 读取数据
data <- read.csv("data.csv")

# 检查数据结构
str(data)
```

数据预处理

- 检查缺失值

```
# 检查缺失值
missing_values <- colSums(is.na(data))
missing_values
```

```
> missing_values
  Observation           Age      Severity      Surg.Med       Anxiety Satisfaction 
            0             0             0             0             0             0 
```

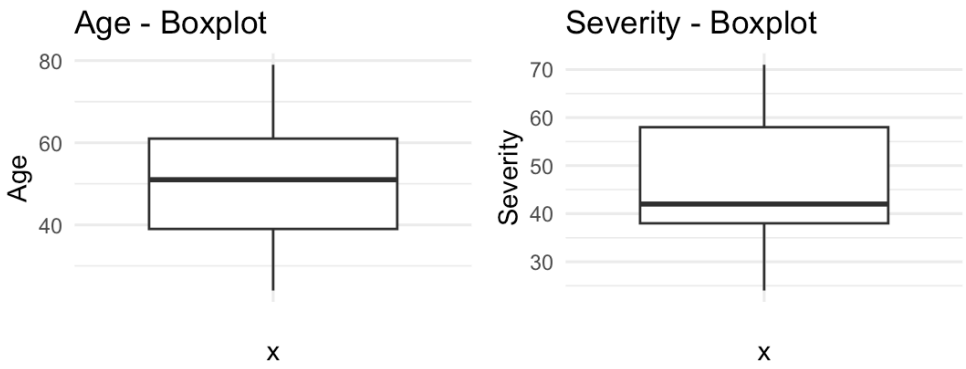
    - 数据集中各个列并无缺失值，无需额外处理

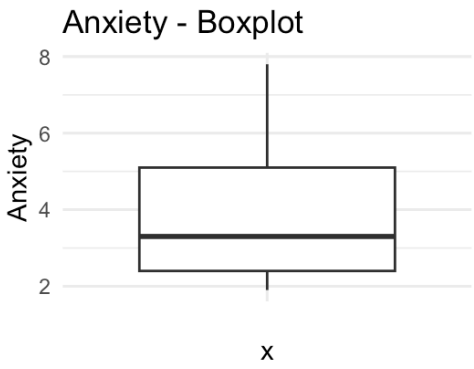
- 检查异常值(离群值)

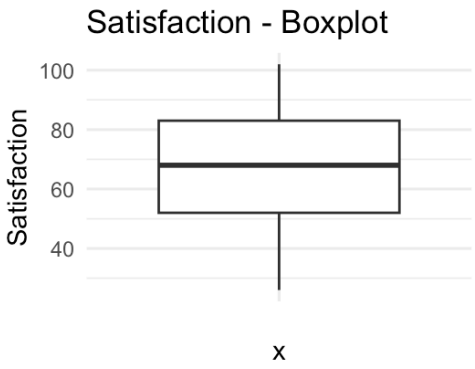
```
# 绘制箱线图来检查异常值
b1 <- ggplot(data, aes(x = "", y = Age)) +
  geom_boxplot() +
  ggtitle("Age - Boxplot") +
  theme_minimal()

b2 <- ggplot(data, aes(x = "", y = Severity)) +
  geom_boxplot() +
  ggtitle("Severity - Boxplot") +
  theme_minimal()

b3 <- ggplot(data, aes(x = "", y = Anxiety)) +
  geom_boxplot() +
```

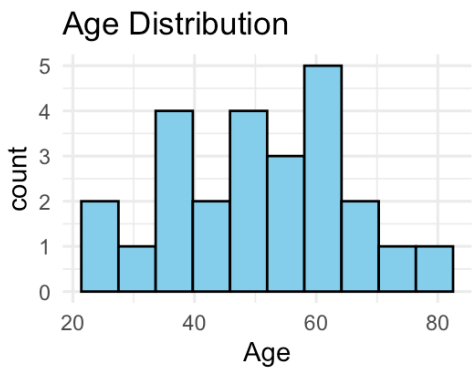
```
  ggtitle("Anxiety - Boxplot") +
  theme_minimal()

b4 <- ggplot(data, aes(x = "", y = Satisfaction)) +
  geom_boxplot() +
  ggtitle("Satisfaction - Boxplot") +
  theme_minimal()

grid.arrange(b1, b2, b3, b4, ncol = 2)

# 绘制各个变量的直方图
p1 <- ggplot(data, aes(x = Age)) +
  geom_histogram(bins = 10, fill = "skyblue", color = "black") +
  ggtitle("Age Distribution") +
  theme_minimal()

p2 <- ggplot(data, aes(x = Severity)) +
  geom_histogram(bins = 10, fill = "lightgreen", color = "black") +
  ggtitle("Severity Distribution") +
  theme_minimal()

p3 <- ggplot(data, aes(x = Anxiety)) +
  geom_histogram(bins = 10, fill = "salmon", color = "black") +
  ggtitle("Anxiety Distribution") +
  theme_minimal()

p4 <- ggplot(data, aes(x = Satisfaction)) +
  geom_histogram(bins = 10, fill = "lightpink", color = "black") +
  ggtitle("Satisfaction Distribution") +
  theme_minimal()

# 将所有图形放置到一个大图中
grid.arrange(p1, p2, p3, p4, ncol = 2)
```

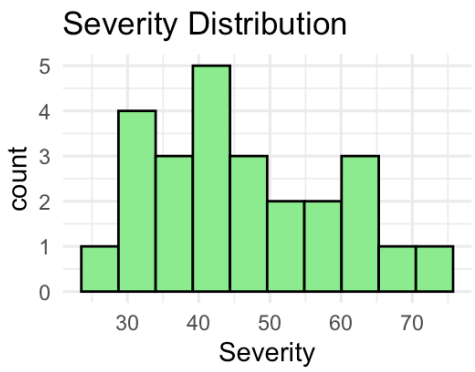
- 结果如下：

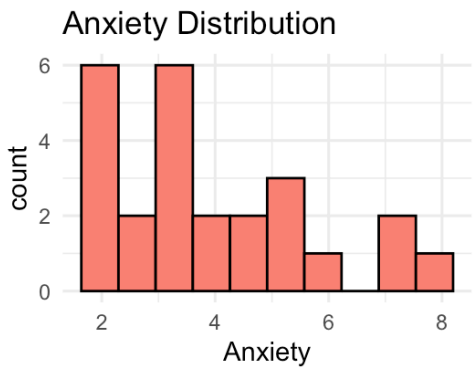


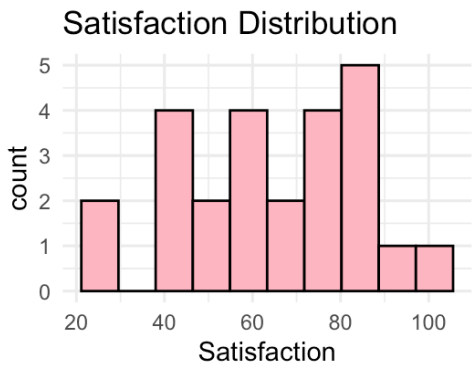












- 经过观察，数据均处于正常范围内，无异常值出现

描述统计分析

```
# 描述性统计分析
summary(data)
```

```
  Observation       Age            Severity        Surg.Med       Anxiety
 Min.   : 1   Min.   :24.00   Min.   :24.00   Min.   :0.00   Min.   :1.900
 1st Qu.: 7   1st Qu.:39.00   1st Qu.:38.00   1st Qu.:0.00   1st Qu.:2.400
```

```
Median :13   Median :51.00   Median :42.00   Median :1.00   Median :3.300
Mean   :13   Mean   :50.84   Mean   :45.92   Mean   :0.56   Mean   :3.932
3rd Qu.:19   3rd Qu.:61.00   3rd Qu.:58.00   3rd Qu.:1.00   3rd Qu.:5.100
Max.   :25   Max.   :79.00   Max.   :71.00   Max.   :1.00   Max.   :7.800
 Satisfaction
Min.   : 26.00
1st Qu.: 52.00
Median : 68.00
Mean   : 65.52
3rd Qu.: 83.00
Max.   :102.00
```

结果初步分析

1. **Age（年龄）**
   - **最小值**为24，**最大值**为79，年龄范围较广，涵盖了年轻人和老年患者。
   - **均值**为50.84，**中位数**为51.00，表明患者的年龄大致集中在50岁左右，且数据呈现出较为对称的分布。
   - 四分位数分析：
     - **第一四分位数（1st Qu.）**为39岁，**第三四分位数（3rd Qu.）**为61岁，说明大部分患者的年龄在39至61岁之间。
       - 年龄分布较为均匀，但可以看到中位数与均值接近，表明年龄没有显著的偏态。
2. **Severity（疾病严重程度）**
   - **最小值**为24，**最大值**为71，疾病的严重程度范围较广，表明样本中既有轻症患者，也有重症患者。
   - **均值**为45.92，**中位数**为42.00，说明大部分患者的病情属于中等严重程度。
   - 四分位数分析：
     - **第一四分位数（1st Qu.）**为38，**第三四分位数（3rd Qu.）**为58，说明数据中有较大比例的患者病情较重（第三四分位数较高）。
       - 这表明患者的病情严重程度分布较为宽泛，可能需要在建模时对疾病严重度变量进行进一步分析。
3. **Surg.Med（内科外科患者指示变量）**
   - 这是一个二元变量（0表示内科患者，1表示外科患者），**均值**为0.56，表示约56%的患者是内科患者，44%为外科患者。
   - 数据中内科和外科患者的比例接近，这可能意味着内科与外科患者的满意度评分差异可能不会特别大，除非其他因素（如焦虑、疾病严重度等）发挥作用。
4. **Anxiety（焦虑指数）**
   - **最小值**为1.9，**最大值**为7.8，焦虑指数的范围显示出患者焦虑程度的差异，且焦虑程度存在一定的分布宽度。
   - **均值**为3.93，**中位数**为3.3，说明大部分患者的焦虑水平属于中等偏低至中等之间。
     - 四分位数分析：
       - **第一四分位数（1st Qu.）**为2.4，**第三四分位数（3rd Qu.）**为5.1，表明焦虑程度的分布存在较大差异，一部分患者焦虑较轻，而另一部分则焦虑较重。

        - 需要注意的是，焦虑程度较高的患者可能会影响他们对医疗服务的满意度，焦虑变量可能对满意度具有较强的影响。

5. **Satisfaction（患者满意度）**
    - **最小值**为26，**最大值**为102，患者满意度的范围从较低的26分到较高的102分，表明患者满意度存在明显差异。
    - **均值**为65.52，**中位数**为68.00，表明大多数患者的满意度处于中等偏上的水平，且均值接近中位数，可能表明患者总体上较为满意。
        - 四分位数分析：
            - **第一四分位数（1st Qu.）**为52，**第三四分位数（3rd Qu.）**为83，说明有一部分患者的满意度较低（低于52），而另一部分则较高（高于83），这一差异可能与其他变量（如焦虑、疾病严重度等）密切相关。

总结

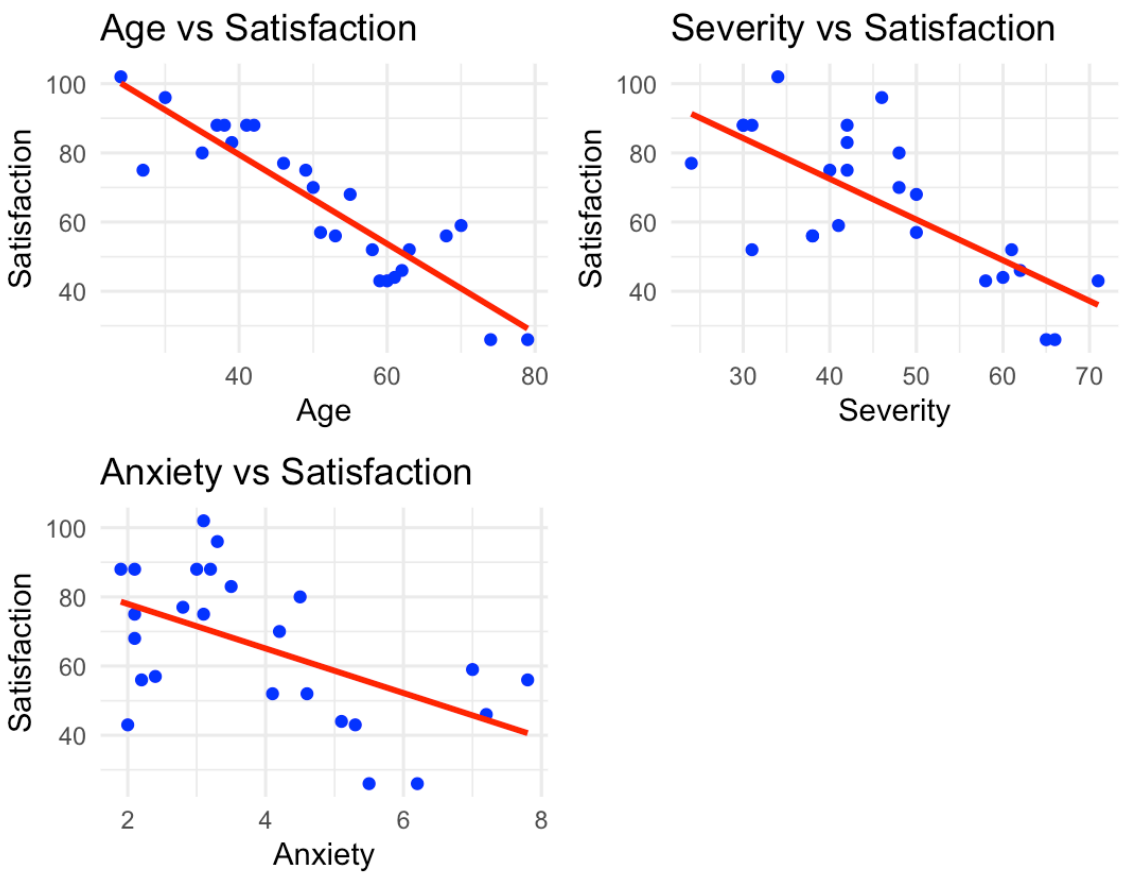
- **年龄**的分布较均匀，患者群体较为多样化。
- **疾病严重度**的分布较广，意味着患者病情的差异性较大，这可能会影响他们的满意度。
- **内科外科患者比例**较为均衡，暗示不同科别的患者群体在建模时可能需要独立考虑。
- **焦虑指数**显示出较大的个体差异，焦虑可能会是影响患者满意度的重要因素，特别是高焦虑的患者可能倾向于对治疗不满。
- **满意度的差异性较大**，既有极低满意度的患者，也有高满意度的患者，这提示在建模时，可能需要特别关注低满意度患者的特征，并探索如何改善他们的就医体验。

3. 建模前准备

初步判断相关关系

```
# 绘制自变量与因变量之间的散点图
p1 <- ggplot(data, aes(x = Age, y = Satisfaction)) +
  geom_point(color = "blue") +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  ggtitle("Age vs Satisfaction") +
  theme_minimal()

p2 <- ggplot(data, aes(x = Severity, y = Satisfaction)) +
  geom_point(color = "blue") +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  ggtitle("Severity vs Satisfaction") +
  theme_minimal()

p3 <- ggplot(data, aes(x = Anxiety, y = Satisfaction)) +
  geom_point(color = "blue") +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  ggtitle("Anxiety vs Satisfaction") +
  theme_minimal()

# 将散点图组合在一起
```

```
grid.arrange(p1, p2, p3, ncol = 2)
```



- 我们观察散点图可以看出，`Age`和`Satisfaction`存在较强的相关关系，`Severity`和`Satisfaction`存在一定的相关关系，`Anxiety`和`Satisfaction`可能存在一定相关关系，那么接下来我们计算相关系数

相关系数计算与热力图

- 这里我们先检验数据是否符合正态分布，然后决定我们使用哪一种相关系数

```
# Shapiro-Wilk 正态性检验
shapiro_test_results <- apply(data[, c("Age", "Severity", "Surg.Med", "Anxiety",
"Satisfaction")], 2, shapiro.test)
shapiro_test_results
```

```
$Age
  Shapiro-Wilk normality test
data:  newX[, i]
W = 0.98189, p-value = 0.9197

$Severity
  Shapiro-Wilk normality test
data:  newX[, i]
W = 0.95634, p-value = 0.3464

$Surg.Med
  Shapiro-Wilk normality test
data:  newX[, i]
```

```
W = 0.63401, p-value = 1.061e-06

$Anxiety
  Shapiro-Wilk normality test
data:  newX[, i]
W = 0.90535, p-value = 0.02404

$Satisfaction
  Shapiro-Wilk normality test
data:  newX[, i]
W = 0.96024, p-value = 0.4193
```

- 观察结果我们看到`Surg.Med`和`Anxiety`，但是`Anxiety`的p值与0.05相比差异并不是很大，因此我们仍然采用皮尔逊相关系数来计算

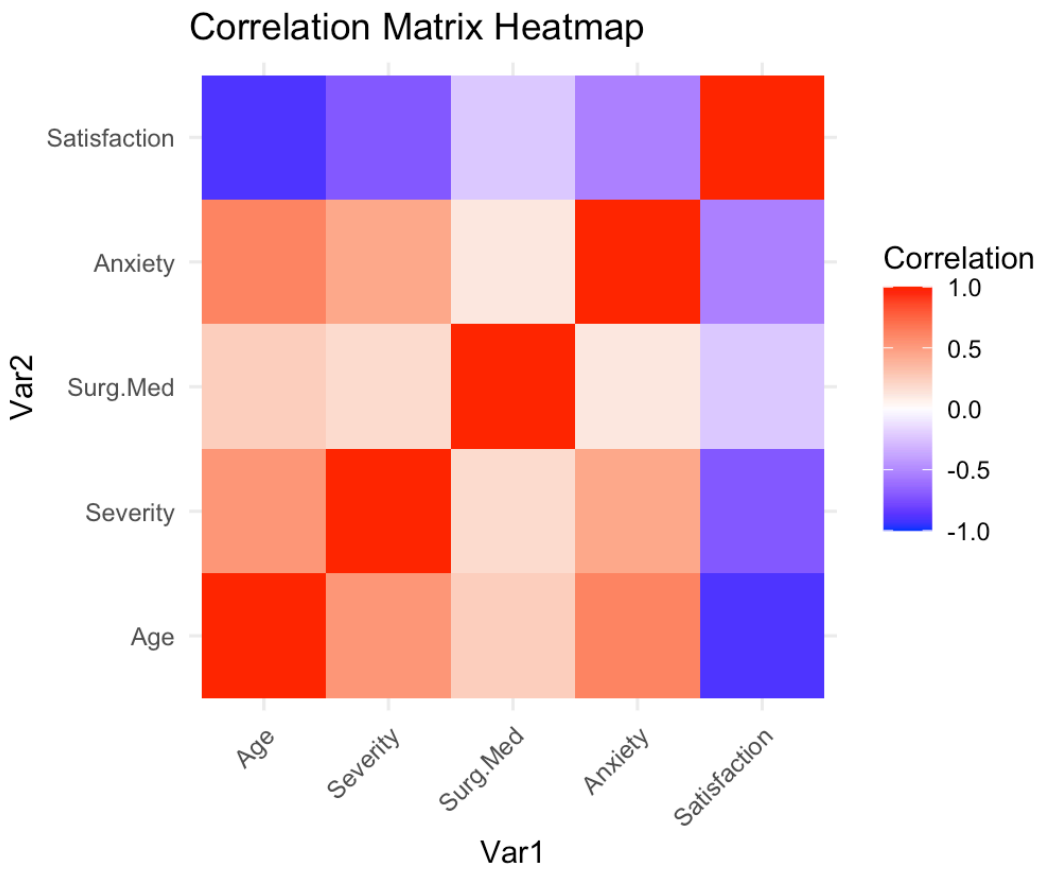
- 相关系数计算与热力图绘制

```r
# 加载相关的包
library(ggplot2)
library(reshape2)

# 计算自变量之间的相关系数矩阵
cor_matrix <- cor(data[, c("Age", "Severity", "Surg.Med", "Anxiety",
"Satisfaction")])

# 将相关系数矩阵转换为适合ggplot的数据格式
cor_melted <- melt(cor_matrix)

# 绘制相关系数矩阵热力图
ggplot(cor_melted, aes(Var1, Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1, 1), name="Correlation") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        axis.text.y = element_text(angle = 0)) +
  labs(title = "Correlation Matrix Heatmap") +
  coord_fixed()
```

  - 相关系数矩阵：

	Age	Severity	Surg.Med	Anxiety	Satisfaction
Age	1.0000000	0.5290246	0.2456932	0.6212453	-0.9013537
Severity	0.5290246	1.0000000	0.1775101	0.4471567	-0.7230139
Surg.Med	0.2456932	0.1775101	1.0000000	0.1096486	-0.2339424
Anxiety	0.6212453	0.4471567	0.1096486	1.0000000	-0.5363177
Satisfaction	-0.9013537	-0.7230139	-0.2339424	-0.5363177	1.0000000

  - 热力图：



- 经过观察我们可以得出：
  - **与Age**的相关系数为 -0.9013537，表示患者的年龄与满意度之间有强烈的负相关关系。年龄越大，满意度越低。
  - **与Severity**的相关系数为 -0.7230139，表明疾病严重程度与满意度也呈显著负相关，严重程度越高，患者的满意度越低。
  - **与Surg.Med**的相关系数为 -0.2339424，显示内科与外科患者的满意度差异较小，相关性较弱。
  - **与Anxiety**的相关系数为 -0.5363177，表明焦虑指数与患者满意度之间有中等程度的负相关，焦虑程度越高，满意度越低。

  结论： 从这些结果可以看出，`Age`和`Severity`对患者满意度有显著的负向影响。年龄和疾病严重度是影响患者满意度的关键因素，而焦虑也有一定的影响，但程度较为中等。内科与外科患者之间的满意度差异较小。

多重共线性检测

- 为什么要进行多重共线性检测?

  多重共线性检测是回归分析中的一个重要步骤，旨在识别自变量之间是否存在高度相关性。若自变量之间高度相关（即多重共线性），会导致回归系数的不稳定和标准误差的膨胀，从而影响模型的解释性和预测能力。具体来说，多重共线性可能使得回归系数的估计变得不准确，模型的显著性检验结果不可靠，且对小的样本变化敏感，难以区分各个自变量的独立贡献。因此，在回归分析中进行多重共线性检测，有助于确保模型的有效性和稳定性，从而得出更可靠的结论。常用的检测方法包括计算方差膨胀因子（VIF），通过VIF值判断自变量间的相关性程度。

- 计算VIF

```r
# 加载car包计算VIF
library(car)

# 构建初步回归模型
model <- lm(Satisfaction ~ Age + Severity + Surg.Med + Anxiety, data = data)

# 计算VIF
vif(model)
```

```
     Age Severity Surg.Med  Anxiety 
1.939128 1.441055 1.072782 1.689768 
```

- 结果分析：

  这里使用了方差膨胀因子（VIF）来检测模型中的多重共线性。VIF的值反映了各个自变量之间的共线性程度：

  - **Age**的VIF为 1.939128，表明与其他自变量有一定的相关性，但这个值小于10，通常认为不存在严重的共线性问题。
  - **Severity**的VIF为 1.441055，同样也小于10，表示与其他自变量的共线性不严重。
  - **Surg.Med**的VIF为 1.072782，非常低，表明它与其他变量几乎没有相关性。
  - **Anxiety**的VIF为 1.689768，同样也低于10，说明它与其他变量的相关性不高。

  结论： 从VIF值来看，该回归模型中不存在明显的多重共线性问题，因此可以继续使用这些自变量进行回归分析。

4. 模型建立

逐步回归

- 使用逐步回归的方式建立多元线性回归模型，并对比三种方式得到的结果

```r
# 加载必要的包
library(MASS)

# 构建初步回归模型（包括所有自变量）
full_model <- lm(Satisfaction ~ Age + Severity + Surg.Med + Anxiety, data = data)

# 进行逐步回归
stepwise_model <- stepAIC(full_model, direction = "both", trace = FALSE)
```

以下是三种逐步回归方法的对比表格：

模型方法	最终变量	R^2	调整后的 R^2	F统计量	p值	解释
双向逐步回归	`Age`, `Severity`	0.8966	0.8872	95.38	< 0.001	该模型简洁，解释力强，去除了不显著的变量，拟合度较高。
前向逐步回归	`Age`, `Severity`, `Surg.Med`, `Anxiety`	0.9036	0.8843	46.87	< 0.001	尽管R^2略高，但 `Surg.Med` 和 `Anxiety` 的 p值较高，不显著，增加了复杂性。
后向逐步回归	`Age`, `Severity`	0.8966	0.8872	95.38	< 0.001	与双向逐步回归相同，简洁且拟合度较好。

关键比较

- **R^2与调整后的 R^2：** 前向逐步回归略优于其他两者，但差异不大。
- **变量显著性：** 双向和后向逐步回归都选择了 `Age` 和 `Severity` 两个显著变量，而前向逐步回归则包括了 `Surg.Med` 和 `Anxiety`，但它们的显著性较低（$p > 0.05$）。
- **模型复杂性：** 双向和后向逐步回归更简洁，避免了包含不显著的变量，容易解释且保持较高的模型质量。

最终选择：双向逐步回归（`stepwise_model`）

- 理由：
  - **模型简洁性与可解释性**：双向逐步回归通过同时考虑添加和删除变量的方式，保留了重要的变量（`Age` 和 `Severity`），剔除了不显著的变量（如 `Surg.Med` 和 `Anxiety`）。这样做的好处是得到一个更加简洁、有效且容易解释的模型。
  - **统计显著性**：双向逐步回归模型的结果表明，`Age` 和 `Severity` 对满意度的预测具有显著性，而且模型的拟合度（R^2 和调整后的 R^2）足够高。
  - **避免不必要的变量**：虽然前向逐步回归模型提高了 R^2，但由于 `Surg.Med` 和 `Anxiety` 对模型的贡献不大，继续保留这些变量可能会增加模型的复杂度，但不一定提高预测能力。

因此，**双向逐步回归**（`stepwise_model`）是最合适的选择，它既能保证模型的预测能力，又保持了较好的简洁性和可解释性。

5. 模型评价

模型拟合优度：R^2 和调整后的 R^2

```
# 1. 模型拟合优度：检查R²和调整后的R²
summary(stepwise_model)
```

```
Call:
lm(formula = Satisfaction ~ Age + Severity, data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-17.2800  -5.0316   0.9276   4.2911  10.4993

Coefficients:
```

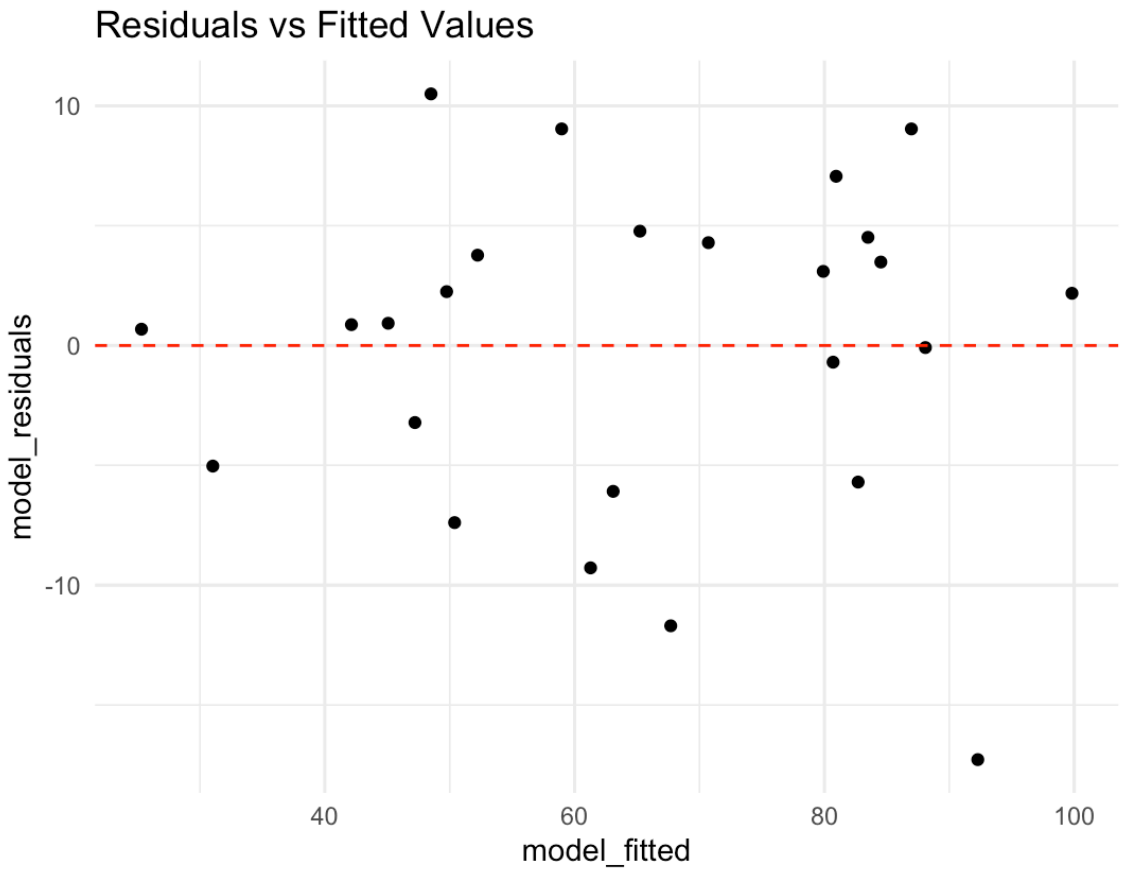
```
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 143.4720     5.9548  24.093  < 2e-16 ***
Age          -1.0311     0.1156  -8.918 9.28e-09 ***
Severity     -0.5560     0.1314  -4.231 0.000343 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 7.118 on 22 degrees of freedom
Multiple R-squared:  0.8966,  Adjusted R-squared:  0.8872
F-statistic: 95.38 on 2 and 22 DF,  p-value: 1.446e-11
```

- **R^2 = 0.8966**：模型解释了约 89.66% 的患者满意度的变化，这表示模型具有很好的拟合度。
- **调整后的 R^2 = 0.8872**：调整后的 R^2考虑了模型中变量的数量，调整后仍然接近原始 R^2，表明模型的解释力保持较好。
- **F统计量 = 95.38，p值 < 0.001**：该结果表明模型的整体拟合非常显著，即 `Age` 和 `Severity` 对患者满意度有显著的影响。

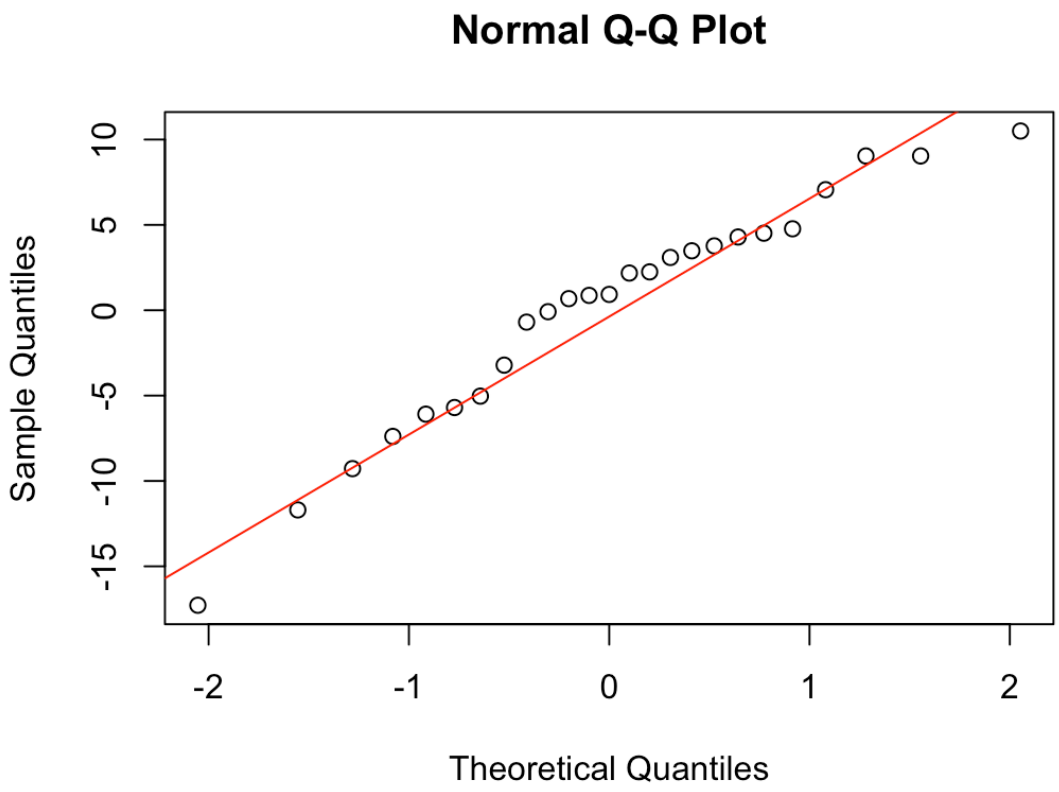
残差分析

```r
# 2. 残差分析
# 提取拟合值和残差
model_residuals <- residuals(stepwise_model)
model_fitted <- fitted(stepwise_model)

# 残差与拟合值的散点图
ggplot(data, aes(x = model_fitted, y = model_residuals)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  ggtitle("Residuals vs Fitted Values") +
  theme_minimal()

# Q-Q图：检验残差的正态性
qqnorm(model_residuals)
qqline(model_residuals, col = "red")

# Shapiro-Wilk正态性检验
shapiro.test(model_residuals)
```

- **残差与拟合值的散点图**：残差随机分布在拟合值周围，没有系统性的模式



- **Q-Q图**：点沿着对角线分布，表明残差符合正态分布。



- **Shapiro-Wilk正态性检验**：

```
  Shapiro-Wilk normality test

data:  model_residuals
W = 0.95367, p-value = 0.3028
```

- **W = 0.95367，p值 = 0.3028**：p值大于0.05，表明没有足够证据拒绝残差正态分布的假设。因此，可以认为残差符合正态分布，这符合回归分析的假设。

方差膨胀因子（VIF）

```
# 3. 计算方差膨胀因子（VIF）来检查多重共线性
library(car)
vif(stepwise_model)
```

```
     Age Severity
1.388632 1.388632
```

- VIF值：
  - `Age` 的 VIF = 1.388632，`Severity` 的 VIF = 1.388632。
  - 这两个 VIF 值远低于常见的阈值 10，表明模型中不存在严重的多重共线性问题。

回归系数显著性

```
summary(stepwise_model)
```

```
Call:
lm(formula = Satisfaction ~ Age + Severity, data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-17.2800  -5.0316   0.9276   4.2911  10.4993

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 143.4720     5.9548  24.093  < 2e-16 ***
Age          -1.0311     0.1156  -8.918 9.28e-09 ***
Severity     -0.5560     0.1314  -4.231 0.000343 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 7.118 on 22 degrees of freedom
Multiple R-squared:  0.8966,  Adjusted R-squared:  0.8872
F-statistic: 95.38 on 2 and 22 DF,  p-value: 1.446e-11
```

- **Age** 的 t 值 = -8.918，p值 < 0.001，说明 `Age` 对满意度的影响非常显著。
- **Severity** 的 t 值 = -4.231，p值 = 0.000343，说明 `Severity` 也对满意度有显著影响。

- **Intercept** 的 p 值 < 0.001，说明截距项也是显著的。

总结

1. **拟合优度**：模型的 R^2 和调整后的 R^2 表明，`Age` 和 `Severity` 这两个变量能够很好地解释患者满意度的变化，且模型整体拟合显著。
2. **残差分析**：残差符合正态分布，且没有明显的异方差性问题。
3. **多重共线性**：VIF值表明模型没有多重共线性问题。
4. **回归系数显著性**：所有重要变量（`Age` 和 `Severity`）的回归系数都具有显著性，模型是有效的。

该回归模型表现出较好的拟合度，残差符合正态性，且不存在多重共线性问题。`Age` 和 `Severity` 对患者满意度的影响是显著的，模型可以有效用于预测患者满意度。

5. 模型未使用变量分析

新模型的回归结果(加入Anxiety)

- 为了防止逐步回归过程中去除掉了其他有用变量，这里将Anxiety加入模型然后检验

```r
# 构建新模型，加入Anxiety变量
new_model <- lm(Satisfaction ~ Age + Severity + Anxiety, data = data)

# 查看新的模型摘要
summary(new_model)
```

```
Call:
lm(formula = Satisfaction ~ Age + Severity + Anxiety, data = data)

Residuals:
     Min       1Q   Median       3Q      Max 
-18.2812  -3.8635   0.6427   4.5324  11.8734 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 143.8952     5.8975  24.399  < 2e-16 ***
Age          -1.1135     0.1326  -8.398 3.75e-08 ***
Severity     -0.5849     0.1320  -4.430 0.000232 ***
Anxiety       1.2962     1.0560   1.227 0.233231    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 7.037 on 21 degrees of freedom
Multiple R-squared:  0.9035,  Adjusted R-squared:  0.8897 
F-statistic: 65.55 on 3 and 21 DF,  p-value: 7.85e-11
```

1. **模型公式**：
   `Satisfaction ~ Age + Severity + Anxiety`
2. **回归系数和显著性**：

- **Intercept (截距):** `143.8952`，p值 < 0.001，显著。
- **Age:** `-1.1135`，p值 = 3.75e-08，显著。
- **Severity:** `-0.5849`，p值 = 0.000232，显著。
- **Anxiety:** `1.2962`，p值 = 0.233231，不显著（p > 0.05）。

3. **模型拟合优度：**
   - **R^2 = 0.9035**：这个新模型解释了约 90.35% 的患者满意度的变化，比之前的 0.8966 略高，表示模型拟合略有改进。
   - **调整后的 R^2 = 0.8897**：调整后的 R^2 表示变量数量调整后的拟合度，虽然略有增加，但比原来的 0.8872 增幅不大。
   - **F统计量 = 65.55，p值 < 0.001**：总体模型仍然显著。

进一步分析 `Anxiety` 变量的作用

- **Anxiety 的显著性：** `Anxiety` 的 p 值为 0.2332，远大于 0.05，因此它在模型中对患者满意度的影响并不显著。这意味着，尽管 `Anxiety` 变量在回归模型中被包括进来，但它并没有提供足够的统计证据，表明它与患者满意度之间有显著的关系。
- **R^2 的变化：** 加入 `Anxiety` 变量后，模型的 R^2 从 0.8966 提高到了 0.9035，表面上看似增加了模型的解释度，但实际上，`Anxiety` 的贡献是微乎其微的。`Anxiety` 的回归系数虽然为正，但并不显著，因此，这一增加的拟合优度可能是由于模型复杂性增加（增加了一个变量）而引起的，并非 `Anxiety` 变量本身对结果有实质性贡献。

结论

尽管加入了 `Anxiety` 变量，模型的拟合度稍有增加，但由于 `Anxiety` 对满意度的影响并不显著（p > 0.05），我们应当考虑将其从模型中移除。以下是几个理由：

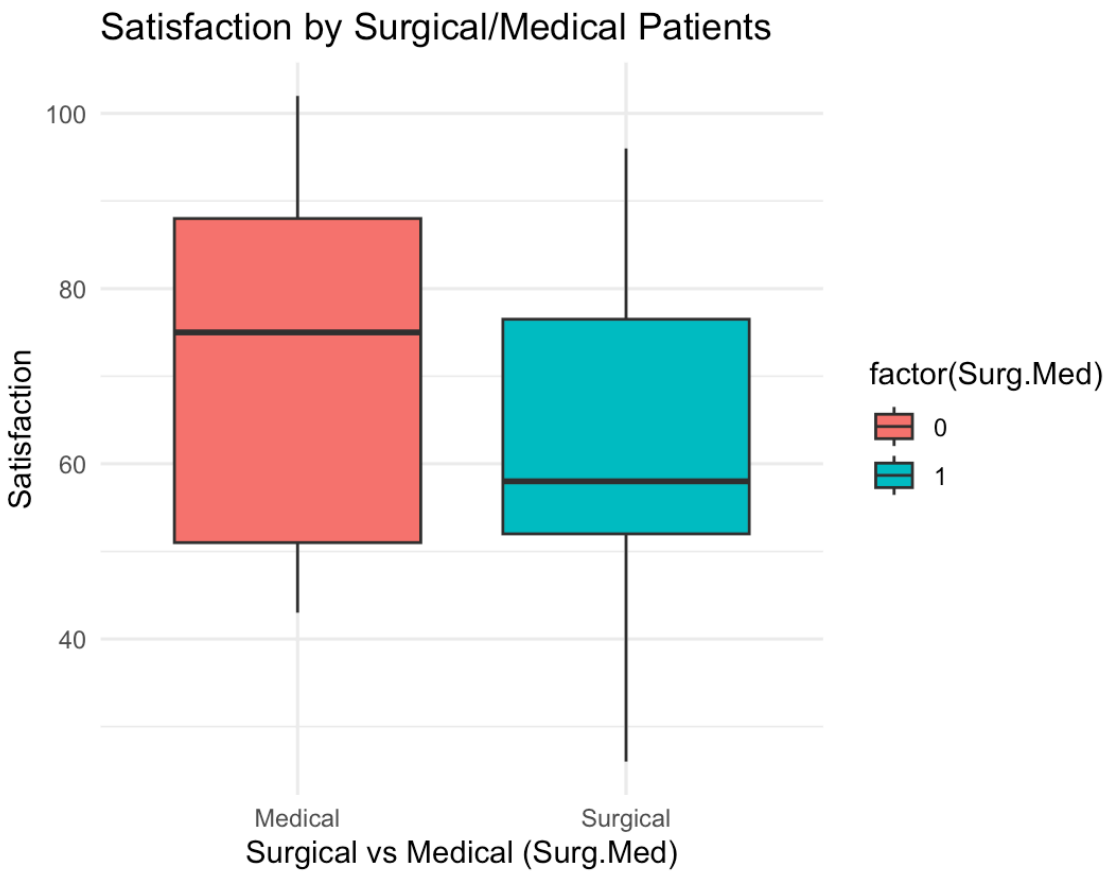
1. **统计显著性不足：** `Anxiety` 的 p 值较大（0.2332），不具备足够的统计显著性，无法证明其对患者满意度的影响。
2. **模型简洁性：** 增加不显著的变量会使模型复杂化，并且可能增加过拟合的风险。简洁的模型通常更易于解释和推理，且可能更具稳定性。
3. **拟合度的微小提升：** 尽管加入 `Anxiety` 后，模型的 R^2 有所提升，但这个提升微不足道，而 `Anxiety` 变量并未显著改善模型预测能力。

因此，**最终建议不选择加入 `Anxiety` 变量**，应保留 `Age` 和 `Severity` 作为模型的主要解释变量。

检验`Surg.Med`作用

```r
# 绘制箱线图，比较内科和外科患者的满意度
ggplot(data, aes(x = factor(Surg.Med), y = Satisfaction, fill = factor(Surg.Med))) +
  geom_boxplot() +
  labs(x = "Surgical vs Medical (Surg.Med)", y = "Satisfaction",
       title = "Satisfaction by Surgical/Medical Patients") +
  scale_x_discrete(labels = c("Medical", "Surgical")) +
  theme_minimal()

# 进行独立样本t检验，判断两组之间的差异
t_test_result <- t.test(Satisfaction ~ Surg.Med, data = data)

# 输出t检验结果
t_test_result
```

```
  Welch Two Sample t-test

data:  Satisfaction by Surg.Med
t = 1.156, df = 21.76, p-value = 0.2602
alternative hypothesis: true difference in means between group 0 and group 1 is not equal
to 0
95 percent confidence interval:
 -7.781348 27.352777
sample estimates:
mean in group 0 mean in group 1
       71.00000        61.21429
```



箱线图分析

- 图形描述：

  箱线图展示了两组患者（内科与外科）满意度的分布情况。

  - **x轴**：代表患者类型（内科与外科），用 `Surg.Med` 变量区分（0 表示内科，1 表示外科）。
  - **y轴**：表示患者的满意度（`Satisfaction`）。
  - **填充颜色**：不同患者类型对应不同颜色。

  从箱线图可以直观地看到，两个组的满意度分布有所不同，内科患者的满意度中位数略高于外科患者。此外，内科患者的满意度范围似乎稍广一些，而外科患者的满意度则有较为集中的趋势。

独立样本 t 检验

接下来，进行 **独立样本 t 检验**，用来判断两组患者（内科与外科）之间是否存在显著的满意度差异。

t 检验结果：

- **t = 1.156**：t 统计量的值，表示样本均值之间的差异相对于样本标准误差的大小。
- **df = 21.76**：自由度，t 检验计算时根据样本量和方差来确定。
- **p-value = 0.2602**：p值为 0.2602，大于常见的显著性水平（0.05）。这表明 **在统计上没有足够的证据** 可以拒绝“两个组之间没有显著差异”的假设。
- **置信区间：** 95% 置信区间为 [-7.78, 27.35]，包含了 0，这也表明两组之间的满意度差异可能为 0。

两组的均值：

- **内科患者的均值（group 0）**：71.00
- **外科患者的均值（group 1）**：61.21

尽管内科患者的平均满意度较外科患者高，但由于 p 值 > 0.05，我们不能得出统计上显著的差异结论。

结论

- 根据 t 检验结果，p 值为 0.2602，表明 **内科和外科患者之间的满意度差异不显著**。
- 即使内科患者的满意度均值略高于外科患者，但这种差异没有达到统计显著性水平。因此，我们无法得出在两个群体之间存在显著满意度差异的结论。

进一步的思考

- **样本量的影响**：在实际情况中，样本量较小可能会导致无法检测到小的差异。可以考虑扩大样本量，看看是否能够找到显著的差异。
- **数据分布**：虽然箱线图显示了两组的分布不同，但从统计学角度来看，这种差异可能只是由于样本的波动，而不具有普遍性。

6. 模型总结

最终回归模型的解读

最终回归模型如下：

$$Satisfaction = 143.472 - 1.031 \times Age - 0.556 \times Severity \quad (1)$$

其中：

- **Intercept (截距): 143.472**
  截距表示当 `Age` 和 `Severity` 都为 0 时，患者的满意度预测值为 143.472。虽然这个值在实际中没有直接意义，但它为我们提供了回归方程的起点。
- **Age (年龄): -1.031**
  `Age` 的回归系数是 -1.031，表示在其他变量（如 `Severity`）保持不变的情况下，年龄每增加 1 单位，患者的满意度将减少 1.031 个单位。这个系数是负的，意味着年龄对患者满意度有负面影响，年龄越大，患者的满意度越低。这个结论可能与某些心理学或医疗研究中的普遍发现一致，即年长患者可能会因健康问题、治疗复杂性等因素而对医疗服务的满意度更低。
- **Severity (疾病严重性): -0.556**
  `Severity` 的回归系数为 -0.556，表示在其他变量（如 `Age`）保持不变的情况下，疾病严重性每增加 1 单位，患者的满意度将减少 0.556 个单位。这个负系数表明疾病的严重程度对患者满意度有显著负向影响，严重的疾病可能导致患者的不满或焦虑，从而影响他们对医疗服务的评价。

不选择其他变量的原因

最终模型中选择了 `Age` 和 `Severity` 作为解释变量，但没有包括其他变量（如 `Surg.Med` 和 `Anxiety`）

1. **`Surg.Med`（患者类型：内科/外科）**：
   - 在 t 检验分析中，`Surg.Med` 对患者满意度的影响并不显著（p 值为 0.2602）。这表明，内科和外科患者在满意度上的差异可能并不大，或者其他因素（如年龄、疾病严重性）在解释满意度差异时已经起到了更重要的作用。因此，尽管 `Surg.Med` 作为分类变量有潜力影响满意度，但在模型中没有显示出足够的显著性，因此被排除在外。
2. **`Anxiety`（焦虑水平）**：
   - 在加入 `Anxiety` 后，回归分析显示该变量的 p 值为 0.233231，远大于 0.05，表明它与患者满意度之间的关系并不显著。虽然 `Anxiety` 对患者的整体健康有一定的影响，但在数据中，它似乎未能显著影响满意度。模型中加入不显著的变量可能会导致模型的复杂度增加，并且增加了过拟合的风险，而不增加模型的解释能力。

总结

- 最终选择的模型包括 `Age` 和 `Severity`，这两个变量在统计上显著，且能够较好地解释患者满意度的变化。
- `Surg.Med` 和 `Anxiety` 被排除的原因是它们与患者满意度之间没有显著的关系（p > 0.05），加入这些变量不会显著提高模型的拟合度，也可能会导致模型复杂度增加，影响模型的稳定性和解释性。

7. 模型优化与改进

未考虑潜在的交互效应

- **问题描述**：在当前的回归模型中，我们假设所有自变量（如`Age`，`Severity`，`Anxiety`等）对患者满意度的影响是独立的。然而，现实情况中，这些自变量之间可能存在交互效应。例如，患者的年龄和焦虑程度可能相互影响，共同作用于患者的满意度。如果没有考虑这些交互效应，可能会导致对因变量的估计不准确或不完全。
- **优化建议**：

- 在模型中添加交互项。例如，可以尝试引入 `Age * Anxiety` 或 `Severity * Anxiety` 等交互项，以考察这些因素是否共同作用于满意度。

样本数量少

- **问题描述**：当前数据集的样本量为25个，样本量相对较小。小样本量可能导致模型不稳定，拟合的结果容易受到异常值或噪声的影响，且在进行假设检验时，容易导致假阳性或假阴性的结果。小样本量还可能限制了变量选择和模型复杂度的增加。
- **优化建议**：
  - **增加样本量**：尝试扩大数据集，收集更多的患者数据，尤其是更多的不同背景或不同特征的患者。这将有助于提高模型的稳健性和预测能力。
  - **交叉验证**：可以使用交叉验证（如K折交叉验证）来提高模型的泛化能力，减少样本量较小带来的偏差。

模型的非线性关系

- **问题描述**：当前使用的回归模型假设自变量与因变量之间的关系是线性的。然而，在实际中，可能存在一些非线性关系，特别是变量如焦虑程度（`Anxiety`）和疾病严重程度（`Severity`）对满意度的影响可能是非线性的。忽略这些非线性关系可能会导致模型拟合不良，无法捕捉到自变量与因变量之间的真实关系。
- **优化建议**：
  - 考虑使用非线性回归模型，如多项式回归、局部加权回归（LOESS），或者使用更复杂的机器学习方法（如随机森林、支持向量机等）来捕捉潜在的非线性关系。
  - 通过绘制自变量与因变量的散点图，检查是否存在明显的非线性关系，如果有，可以尝试对数据进行转换（如对数转换、平方项等）。

二、分类模型

1. 项目背景

- 随着企业竞争的日益加剧，员工的留存和离职问题已经成为人力资源管理中的一个重要课题。员工离职不仅会造成企业运营的中断和人才流失，还可能增加招聘和培训新员工的成本。因此，能够准确预测员工离职的因素，并为人力资源部门提供有效的管理建议，已经成为现代企业人力资源管理中的关键需求之一。
- 本项目旨在通过对公司员工数据的分析，帮助人力资源部门洞察员工离职的潜在原因，评估现有薪资结构、工作环境以及员工背景对离职行为的影响，为企业制定更加精准的人力资源管理策略提供数据支持。数据集包括了员工的教育背景、工作经验、城市分布、薪资等级等多个变量，这些特征可以揭示员工离职的潜在模式和趋势。通过应用统计分析与机器学习技术，项目不仅关注员工离职的现状和原因，还探索了不同员工群体的特征分布和相关性，进一步为企业提供有效的优化建议。
- 在任务方面，本项目首先通过数据探索分析，了解员工的基本情况和不同维度之间的关系；其次，通过构建贝叶斯网络和逻辑回归模型，预测员工是否可能离职，并对模型结果进行解读，分析影响员工离职的主要因素；最后，识别模型中可能存在的不足之处，并提出相应的优化建议，以期提高模型的预测准确度和可解释性，帮助企业更好地管理员工留存。

2. 数据处理

- 在进行数据分析前，我们首先导入数据并进行基本处理，包括数据加载、清洗、标准化等。

数据加载与导入

- 首先，我们加载了名为 `employee.csv` 的数据集。该数据集包含员工的基本信息，如教育背景、工作经验、薪资级别等。我们使用 `read.csv()` 函数导入数据：

```
employee_data <- read.csv("employee.csv")
```

变量类型转换

- 为了确保数据的准确性和适用性，我们对数据集中的多个变量进行了类型转换。具体操作如下：
  - **教育背景（Education）**：该变量表示员工的学历层级。由于学历变量具有顺序性（从本科到硕士，再到博士），我们将其转换为有序因子（`ordered = TRUE`），并指定了学历的顺序。

    ```
    employee_data$Education <- factor(employee_data$Education,
                                      levels = c("Bachelors", "Masters", "PHD"),
                                      ordered = TRUE)
    ```

- **城市（City）**：该变量表示员工所在的工作城市。我们将其转换为无序因子（`as.factor()`）。

  ```
  employee_data$City <- as.factor(employee_data$City)
  ```

- **薪资级别（PaymentTier）**：该变量表示员工的薪资层级。由于薪资级别有明确的顺序（例如 1 级为最低，3 级为最高），我们将其转换为有序因子。

  ```
  employee_data$PaymentTier <- factor(employee_data$PaymentTier,
                                      levels = c(1, 2, 3),
                                      ordered = TRUE)
  ```

- **性别（Gender）**：该变量表示员工的性别。我们将其转换为无序因子，以便进行分析。

  ```
  employee_data$Gender <- as.factor(employee_data$Gender)
  ```

- **是否曾经待岗（EverBenched）**：该变量表示员工是否曾有过待岗状态，转换为有序因子。

  ```
  employee_data$EverBenched <- factor(employee_data$EverBenched,
                                      levels = c("No", "Yes"),
                                      ordered = TRUE)
  ```

- **是否离职（LeaveOrNot）**：该变量表示员工是否已离职。我们将其转换为因子，便于后续的分类分析。

  ```
  employee_data$LeaveOrNot <- as.factor(employee_data$LeaveOrNot)
  ```

- **连续变量（JoiningYear, Age, ExperienceInCurrentDomain）**：由于这些变量是数值型的（年份、年龄和工作经验），我们将其转换为数值型变量。

```
employee_data$JoiningYear <- as.numeric(employee_data$JoiningYear)
employee_data$Age <- as.numeric(employee_data$Age)
employee_data$ExperienceInCurrentDomain <-
as.numeric(employee_data$ExperienceInCurrentDomain)
```

缺失值处理

- 为了提高数据质量和确保分析的可靠性，我们对数据集中的缺失值进行了处理。由于数据量足够，我们对于存在缺失项目的数据，采用简单的删除操作：

```
employee_data <- na.omit(employee_data)  # 删除缺失值的行
```

3. 数据初步分析

- 我们首先分析并回答以下三个问题：
  - 员工的学历背景分布是怎样的?
  - 不同城市的员工服务年限有何差异？差异是否显著?
  - 薪资等级与当前领域经验之间是否存在某种关联?

准备工作

- 我们首先加载所需库

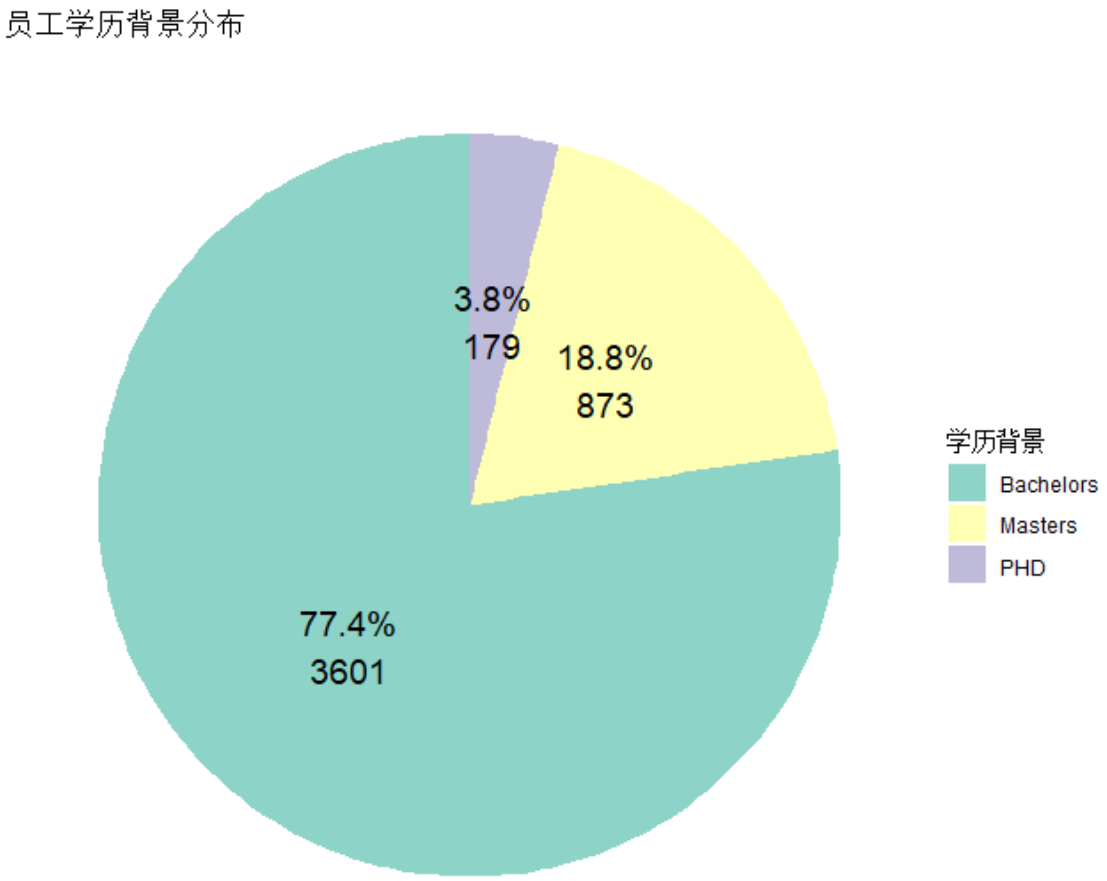
```
library(tidyverse)
```

学历分布

- 接下来，我们首先研究***员工的学历背景分布***。
- 我们可以通过***饼状图***，简单的观察出雇员的学历分布情况。代码如下

```
# 1.1 学历背景分布：饼状图
ggplot(employee_data, aes(x = "", fill = Education)) +
  geom_bar(width = 1, stat = "count") +
  coord_polar(theta = "y") +
  ggtitle("员工学历背景分布") +
  theme_void() +
  theme(axis.text.x = element_blank()) +
  labs(fill = "学历背景") +
  scale_fill_brewer(palette = "Set3") +
  geom_text(aes(label = paste0(round((..count..)/sum(..count..)*100, 1), "%\n",
..count..)),
            stat = "count", position = position_stack(vjust = 0.5), size = 5)
```

- 所得饼状图如下



- 可以看出：

  在该公司雇员中，本科学历背景占绝对主导；同时存在一定的硕士背景；博士学历背景极为少见。 (2)

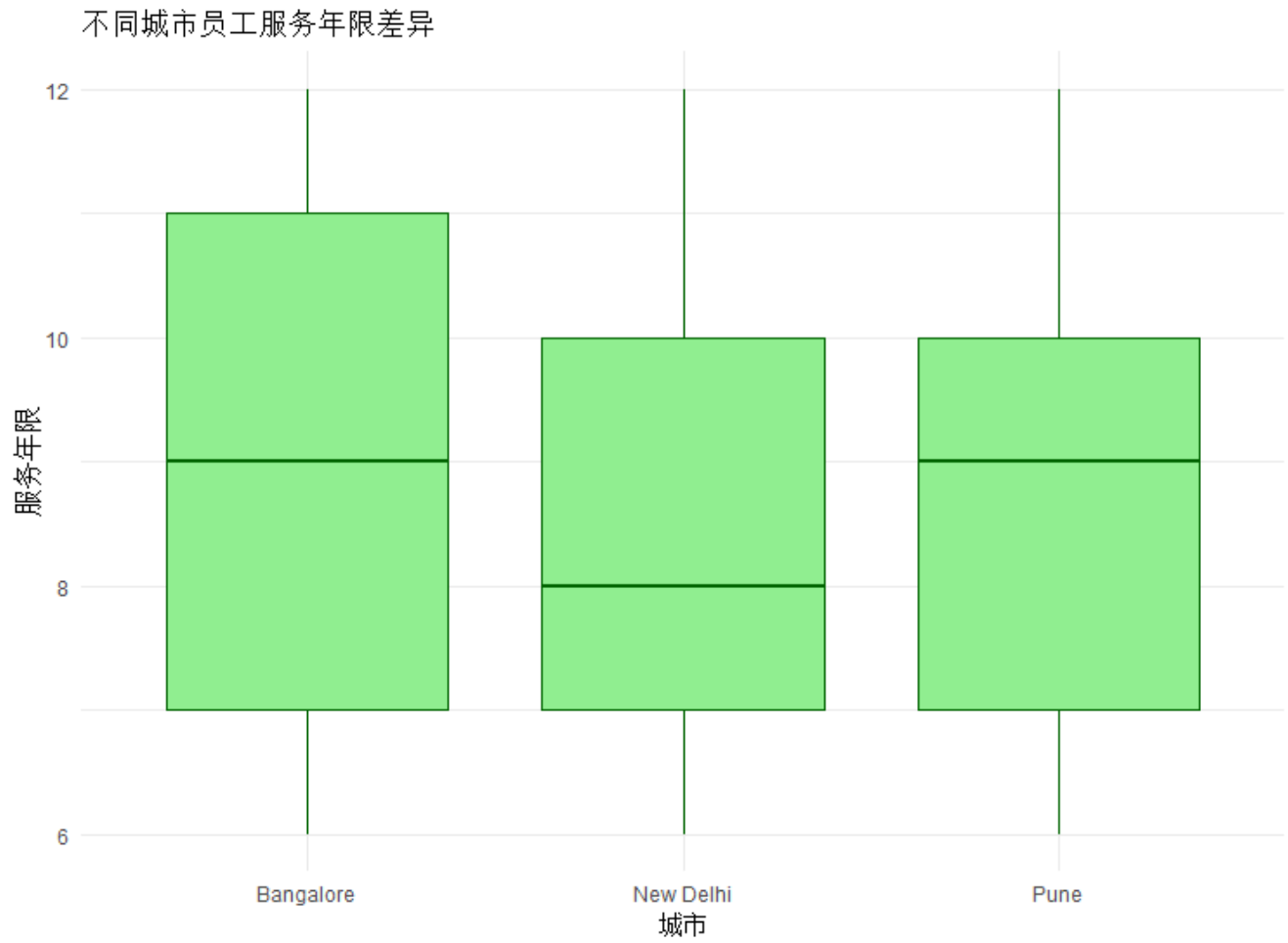
- 这一结论也符合常态。

服务年限分析

- 我们随后分析***不同城市的员工服务年限有何差异，及差异是否显著***。
- 我们可以通过***箱线图***，简单的观察出不同城市的员工服务年限有何差异。代码如下

```
# 1.2 不同城市的员工服务年限差异（箱线图）
# 计算服务年限
employee_data$ServiceYears <- 2024 - employee_data$JoiningYear
ggplot(employee_data, aes(x = City, y = ServiceYears)) +
  geom_boxplot(fill = "lightgreen") +
  ggtitle("不同城市的员工服务年限差异") +
  xlab("城市") +
  ylab("服务年限") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

- 所得图片如下



可以较为直观的看出

$$\text{不同城市的雇员，服务年限存在明显差异。} \tag{3}$$

- 我们随后使用***方差分析***，定量分析差异存在与否。代码如下

```
# 进行ANOVA（方差分析）检验不同城市间服务年限的差异是否显著
anova_result <- aov(ServiceYears ~ City, data = employee_data)
summary(anova_result)
```

- 所得结果为

```
             Df Sum Sq Mean Sq F value Pr(>F)
City          2    341   170.4   50.12 <2e-16 ***
Residuals  4650  15812     3.4
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

- 我们可以观察到，`p-value` 远小于 0.05。据此，我们有充分的理由认为

$$\text{不同城市的雇员，服务年限存在差异，且差异十分显著} \tag{4}$$

薪资等级与当前领域经验关联分析

- 我们随后分析***薪资等级与当前领域经验是否存在关联***。
- 我们首先使用**方差分析**，判断不同当前领域经验条件下，雇员薪资等级是否存在显著差异。

```
# 进行方差分析，检验不同薪资等级之间的领域经验差异是否显著
anova_result <- aov(ExperienceInCurrentDomain ~ PaymentTier, data = employee_data)
summary(anova_result)
```

- 结果如下

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
PaymentTier	2	6	2.765	1.139	0.32
Residuals	4650	11290	2.428		

- 可以看出，`p-value` 远大于 0.05，不同当前领域经验条件下，雇员薪资等级不存在显著差异。因此，我们可以简单直接的认为，

$$\text{薪资等级与当前领域经验不存在关联。} \tag{5}$$

4. 贝叶斯网络模型

- 接下来，我们将通过构建贝叶斯网络模型预测员工是否离职，并评估模型质量。

准备工作

- 我们首先加载所需库

```
library(tidyverse)
```

选择用于构建模型的特征

- 初始的，我们首先选择数据集中的所有特征来构建预测模型：

```
employee_data_subset <- employee_data
```

数据预处理与分割

- 随后，我们对数据进行了分割，将70%的数据用作训练集，剩余的30%用于测试集。为了确保每次实验的结果可复现，我们设置了随机种子。

```
# 数据分割（70% 训练集，30% 测试集）
set.seed(123)  # 设置随机种子以保证结果可复现
train_indices <- sample(1:nrow(employee_data_subset), size = 0.7 *
nrow(employee_data_subset))
train_data <- employee_data_subset[train_indices, ]
test_data <- employee_data_subset[-train_indices, ]
```

贝叶斯网络模型构建与训练

- 随后，我们使用 **贝叶斯网络** 来构建预测模型。在这里，我们选择了 **`bnlearn`** 包中的 **`hc`** 算法来构建模型，并对训练数据进行学习。

```
# 使用训练数据学习贝叶斯网络
model <- bnlearn::tabu(train_data)
fitted_model <- bnlearn::bn.fit(model, data = train_data)
# 查看拟合后的模型
print(fitted_model)
```

模型预测

- 在构建完贝叶斯网络后，我们使用该模型对测试数据进行预测，查看预测结果。具体而言，我们预测了`LeaveOrNot`（员工是否离职）这一目标变量。

```
# 对测试数据进行预测
predictions <- predict(fitted_model, test_data, node = "LeaveOrNot")

# 查看预测结果
head(predictions)
```

模型性能评估

- 然后，我们使用**混淆矩阵**来评估模型的性能。混淆矩阵可以帮助我们分析预测值与真实值之间的关系，并计算模型的准确率、精度、召回率和F1分数。

```
# 计算混淆矩阵
conf_matrix <- table(Predicted = predictions, Actual = test_data$LeaveOrNot)

# 计算模型的准确率、精度、召回率和 F1 分数
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
precision <- conf_matrix[2, 2] / sum(conf_matrix[2, ])
recall <- conf_matrix[2, 2] / sum(conf_matrix[, 2])
f1_score <- 2 * (precision * recall) / (precision + recall)

# 打印评估结果
cat("Accuracy:", accuracy, "\n")
cat("Precision:", precision, "\n")
cat("Recall:", recall, "\n")
cat("F1 Score:", f1_score, "\n")
```

```
> # 打印评估结果
> cat("Accuracy:", accuracy, "\n")
Accuracy: 0.7428367
> cat("Precision:", precision, "\n")
Precision: 0.7241379
> cat("Recall:", recall, "\n")
Recall: 0.4294479
> cat("F1 Score:", f1_score, "\n")
F1 Score: 0.5391528
```

- 模型评估结果显示，贝叶斯网络在预测员工是否离职方面的准确率为74.3%，精度为72.4%，召回率为42.9%，F1分数为53.9%。虽然**准确率较高，表明模型整体预测能力较强，但召回率较低**，意味着模型未能识别出足够多的离职员工。较低的召回率可能导致企业错失提前干预的机会，增加员工流失风险。尽管模型在预测离职员工时具有较好的精度，但为了提高整体预测效果，还应注意提升召回率。
- **综合来看，当前模型能够提供一定的预测能力，但仍有优化空间。**

5. 逻辑回归模型

- 接下来，我们将构建逻辑回归模型，对员工是否会离职进行预测，并通过模型解读员工离职的影响因素。

数据预处理与分割

- 首先，我们对数据进行了预处理，并将其分割为训练集和测试集。同样的，我们使用 **70%** 的数据作为训练集，**30%** 的数据作为测试集，以评估模型的泛化能力。为了确保每次实验的结果可复现，我们设置了随机种子。

```
# 数据分割（70% 训练集，30% 测试集）
set.seed(123)  # 设置随机种子以保证结果可复现
train_indices <- sample(1:nrow(employee_data), size = 0.7 * nrow(employee_data))
train_data <- employee_data[train_indices, ]
test_data <- employee_data[-train_indices, ]
```

逻辑回归模型构建与训练

- 在数据准备完毕后，我们使用 **逻辑回归模型**（Logistic Regression）进行员工是否离职的预测。

```
# 构建逻辑回归模型
logit_model <- glm(LeaveOrNot ~ Education + City + PaymentTier + Age +
ExperienceInCurrentDomain + Gender + EverBenched + JoiningYear,
                   data = train_data, family = binomial)

# 查看模型摘要
summary(logit_model)
```

模型预测

- 模型训练完成后，我们使用逻辑回归模型对测试集数据进行预测，并计算员工离职的概率值。根据这些概率值，我们将员工分类为“离职”或“未离职”。我们设置了 **0.3** 的阈值来进行分类，表示当预测概率大于 **0.3** 时，员工被预测为离职。

```
# 对测试集进行预测（预测离职的概率）
pred_probs <- predict(logit_model, test_data, type = "response")

# 将概率转换为预测标签（阈值0.3）
predictions <- ifelse(pred_probs > 0.3, 1, 0)
```

模型评估

- 接下来，我们使用 **混淆矩阵** 来评估模型的性能。混淆矩阵可以帮助我们分析预测值与真实值之间的关系，并计算模型的准确率、精度、召回率和 F1 分数。

```
# 计算混淆矩阵
conf_matrix <- table(Predicted = predictions, Actual = test_data$LeaveOrNot)

# 打印混淆矩阵
print(conf_matrix)

# 计算准确率、精度、召回率和 F1 分数
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
precision <- conf_matrix[2, 2] / sum(conf_matrix[2, ])
recall <- conf_matrix[2, 2] / sum(conf_matrix[, 2])
f1_score <- 2 * (precision * recall) / (precision + recall)

# 打印评估结果
cat("Accuracy:", accuracy, "\n")
cat("Precision:", precision, "\n")
cat("Recall:", recall, "\n")
cat("F1 Score:", f1_score, "\n")
```

- 输出结果为：

```
> # 打印评估结果
> cat("Accuracy:", accuracy, "\n")
Accuracy: 0.6812321
> cat("Precision:", precision, "\n")
Precision: 0.5317003
> cat("Recall:", recall, "\n")
Recall: 0.7546012
> cat("F1 Score:", f1_score, "\n")
F1 Score: 0.6238377
```

- 可以看到，逻辑回归模型在预测员工是否离职方面的准确率为 **68.1%**，精度为 **53.2%**，召回率为 **75.5%**，F1 分数为 **62.4%**。虽然**召回率较高**，表明模型能够识别出大部分的离职员工，帮助企业及时干预以减少流失风险，但**精度较低**，意味着模型会预测一些并不离职的员工为离职，可能导致过多的误报（假阳性），并产生不必要的干预成本。
- 尽管召回率较高，能够有效捕捉到大部分离职员工，精度较低表明在模型的优化过程中还需考虑减少误报的情况。

综合来看，当前模型具有较好的离职预测能力，但在精度和召回率之间需要找到更合适的平衡，进一步优化以提升其在实际应用中的效果。

模型系数解读

- 通过查看逻辑回归模型的系数，我们可以理解哪些特征对员工离职的预测影响较大。以下是模型的系数（输出是每个特征的权重）：

```
# 模型系数解读
summary(logit_model)
```

- 输出结果为

```
Call:
glm(formula = LeaveOrNot ~ Education + City + PaymentTier + Age +
    ExperienceInCurrentDomain + Gender + EverBenched + JoiningYear,
    family = binomial, data = train_data)

Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)               -3.878e+02  4.620e+01  -8.395  < 2e-16 ***
Education.L                9.011e-02  1.567e-01   0.575  0.56528
Education.Q               -4.834e-01  1.179e-01  -4.099 4.15e-05 ***
CityNew Delhi             -5.418e-01  1.173e-01  -4.620 3.84e-06 ***
CityPune                   5.115e-01  1.024e-01   4.995 5.89e-07 ***
PaymentTier.L             -1.432e-01  1.223e-01  -1.171  0.24180
PaymentTier.Q             -5.206e-01  1.051e-01  -4.953 7.31e-07 ***
Age                       -3.554e-02  8.534e-03  -4.165 3.12e-05 ***
ExperienceInCurrentDomain -6.937e-02  2.618e-02  -2.650  0.00806 **
GenderMale                -8.632e-01  8.502e-02 -10.153  < 2e-16 ***
EverBenched.L              4.135e-01  8.895e-02   4.648 3.35e-06 ***
JoiningYear                1.932e-01  2.293e-02   8.427  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 4180.5  on 3256  degrees of freedom
Residual deviance: 3679.7  on 3245  degrees of freedom
AIC: 3703.7

Number of Fisher Scoring iterations: 4
```

- 对此，我们可以观察到，在模型中，教育背景的线性项（Education.L）的系数为0.0901，且p值为0.56528，表示教育背景对离职的影响不显著。而教育背景的二次项（Education.Q）系数为-0.4834，且p值小于0.05，表明教育背景的变化对离职有显著的非线性影响。城市因素方面，选择新德里（CityNew Delhi）为基准城市时，系数为-0.5418，且p值非常小，表明新德里的员工比其他城市的员工更不容易离职；而在普纳（CityPune）工作的员工更容易离职，其系数为0.5115，p值也很小。关于薪酬等级，薪酬等级的线性项（PaymentTier.L）对离职没有显著影响（p值为0.24180），但薪酬等级的二次项（PaymentTier.Q）显著影响离职（系数为-0.5206，p值小于0.05），暗示薪酬等级的变化对离职的影响是非线性的。
- 此外，年龄（Age）对离职的影响显著，系数为-0.0355，说明年龄较大的员工离职的可能性较低。当前领域的经验（ExperienceInCurrentDomain）对离职有显著负面影响，经验越多，离职的可能性越低，系数为-0.0694。性别方面，男性员工比女性员工更不容易离职，系数为-0.8632，且p值非常小，表明性别对离职有显著影响。曾经被“bench”过的员工（EverBenched）离职的可能性较高，系数为0.4135，且p值很小。最后，入职年份（JoiningYear）对离职有显著影响，系数为0.1932，表明入职较早的员工离职的几率较低。
- 总的来说，我们可以得出如下结论：

1. **城市**因素，特别是**Pune**的员工离职几率较高，而**New Delhi**的员工离职几率较低。

2. **性别**，男性员工比女性员工更不容易离职。
3. **被bench**的员工离职的可能性更高。
4. **年龄**和**经验年数**对离职有负面影响，年龄和经验较多的员工离职的可能性较低。
5. **入职年份**也显著影响员工离职，较早入职的员工离职的几率较低。

6. 模型潜在不足与优化建议

- 尽管通过贝叶斯网络和逻辑回归模型，我们能够对员工离职进行一定程度的预测，但仍然存在一些潜在的不足，且模型的表现可以进一步优化。以下是针对当前模型的潜在不足和相应的优化建议：

模型潜在不足与对应的优化

1. **召回率与精度的平衡问题**
    - 在贝叶斯网络模型中，虽然准确率较高（74.3%），但召回率较低（42.9%），说明模型未能识别出足够多的离职员工。这会导致企业错失一些可能离职的员工，增加流失风险。类似地，逻辑回归模型虽然召回率较高（75.5%），但精度较低（53.2%），即模型预测了许多并未离职的员工为离职员工，可能导致不必要的干预和成本。
    - **优化建议**：我们可以通过调整模型的决策阈值来平衡精度和召回率。例如，调整逻辑回归模型的分类阈值（比如提高或降低预测为“离职”员工的概率阈值），以提高召回率的同时减少误报，或考虑使用更加复杂的评估指标如ROC曲线和AUC值来评估模型的综合表现。
2. **特征选择和模型复杂度**
    - 目前使用的特征较多，且大部分变量的相关性较低。模型可能会因此出现过拟合的现象，尤其是一些不重要的特征可能影响模型的表现。尤其是在逻辑回归模型中，某些变量如教育背景、薪资等级等的影响并不显著，可能不应出现在模型中。
    - **优化建议**：采用特征选择方法（如L1正则化、递归特征消除等）来减少模型的复杂度，筛选出最具预测力的特征，避免无关特征的干扰，降低过拟合的风险。可以通过逐步回归等方法进一步优化特征。
3. **数据不平衡问题**
    - 数据集中“离职”与“未离职”标签的分布可能存在不平衡问题。若“离职”员工远少于“未离职”员工，模型可能会偏向预测“未离职”类别，从而导致较低的召回率和预测准确性。
    - **优化建议**：可以采用处理不平衡数据的方法，如过采样（如SMOTE技术）或欠采样，或者使用类加权的损失函数，使得模型在训练过程中对少数类（离职员工）给予更大的权重。
4. **缺失数据的处理方式**
    - 当前的缺失数据处理方法是直接删除包含缺失值的记录，这可能导致数据丢失，影响模型的训练效果。虽然数据集较大，但仍然有可能导致信息损失。
    - **优化建议**：可以考虑更复杂的缺失值填充方法，如基于回归模型或最近邻的插补方法，来保留更多的信息，而非简单删除缺失值。

进一步的优化策略

1. **模型集成与融合**

    - 可以尝试通过模型集成方法（如随机森林、梯度提升机等）将多个基础模型的预测结果进行组合，以提高模型的稳定性和泛化能力。例如，使用Bagging或Boosting方法结合多个弱学习器（如决策树）可以有效降低模型的偏差和方差，提升预测性能。
2. **时间序列特征的引入**
    - 如果数据集中包含员工的历史行为数据（如历年离职率、晋升记录等），可以考虑将时间序列特征引入模型，增强模型的动态预测能力。历史行为通常对员工离职具有很强的预测作用。
3. **深度学习的探索**
    - 如果数据量较大且数据特征较复杂，深度学习模型（如神经网络）可能能够捕捉到更深层次的特征关联。在此基础上，使用深度学习技术进行特征自动提取和关系建模可能会有更好的预测效果。

三、时序模型

1. 引言

项目背景

- 房价是社会经济活动的重要指标，直接影响着房地产市场的健康发展及购房者的决策。基于时间序列分析的房价预测可以帮助房地产中介优化营销策略，为购房者提供投资建议，同时为政策制定者提供数据支撑。本项目以某特定区域2007年至2019年间的房产销售数据为基础，通过分析房价的历史变化规律，运用统计和建模方法对未来12个月的房价进行预测。

项目目标

- 本项目旨在通过对房产销售数据的深入分析，建立合适的时间序列模型（如 AR、MA、ARMA 或 ARIMA），实现对房价的准确预测。具体目标包括：
    - 分析和可视化不同房产类型在不同时间段的价格变化趋势；
    - 评估和验证时间序列数据的平稳性及随机性；
    - 通过建模与评估，确定最佳模型并进行未来12个月的房价预测；
    - 针对模型结果提出对房地产中介和购房者的实用性建议；
    - 识别模型的局限性并提出改进方向。

报告结构

- 本报告首先对数据进行探索性分析，包括数据加载、清洗和可视化；随后，完成时间序列的预处理及平稳性检验；接着，运用多种统计模型对数据进行拟合和预测，并评估模型质量；最后，对结果进行解读，并提出实际建议和优化方案。报告的最后部分对本项目的主要结论进行总结，并展望后续研究方向。

2. 数据分析

数据加载与检查

- 为了分析和预测房价，本项目首先从提供的数据集 `sales.csv` 中加载了包含销售日期、价格、房产类型、卧室数量及邮政编码的五个字段。通过检查数据的结构和完整性，确保分析和建模的基础数据可靠。

（1） **数据加载与转换**：使用 R 语言读取 sales.csv 数据。转换销售日期 datesold 为标准日期格式，方便时间序列分析。
（2） **数据完整性检查**：检查是否存在缺失值、重复值及异常数据。对价格字段（price）进行基本统计描述（均值、中位数、标准差等）。

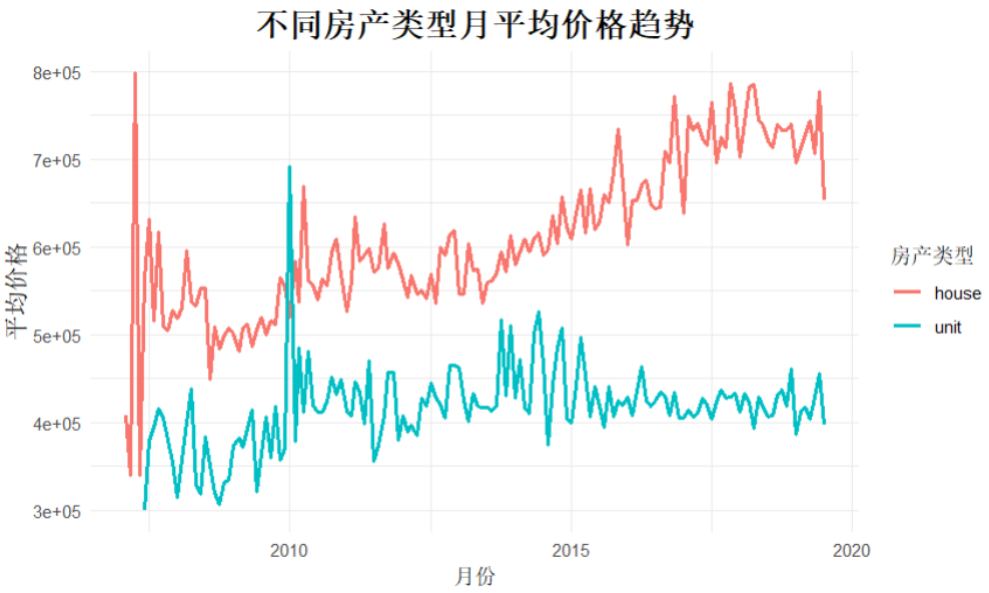
数据可视化

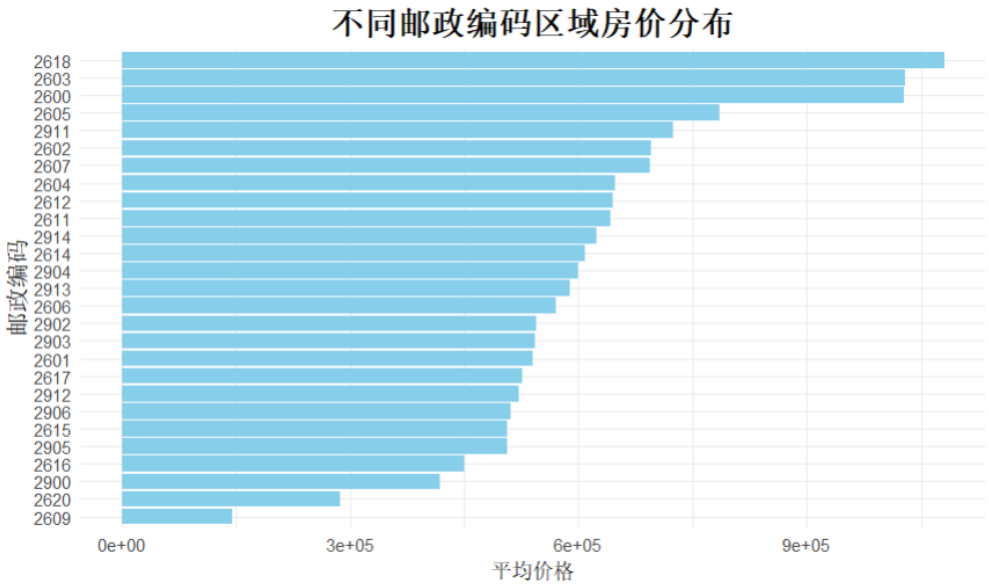
- 通过对数据的可视化分析，探索不同维度上的房价变化规律，提供模型建模的重要参考。
  1. **房产类型的月平均价格趋势：**
  - 按月汇总销售价格，并按房产类型（propertyType）分组。
  - 绘制时间序列折线图，展示各类房产月平均价格的波动趋势。
  - 发现房产类型间价格差异显著，其中 House 类型的价格整体高于 Unit 类型。
  1. **邮政编码区域的房价分布：**
  - 计算每个邮政编码（postcode）的平均房价。
  - 绘制条形图展示各区域房价差异，直观反映房产价格的空间分布。
  - 不同区域的房价差异明显，可能与地段、政策或经济条件相关。
  1. **房价与卧室数量关系：**
  - 使用箱线图展示不同卧室数量（bedrooms）下的房价分布。
  - 数据显示，卧室数量对价格影响较大，但也存在一定的异常点（如大卧室房价低）。
- 相关代码如下：

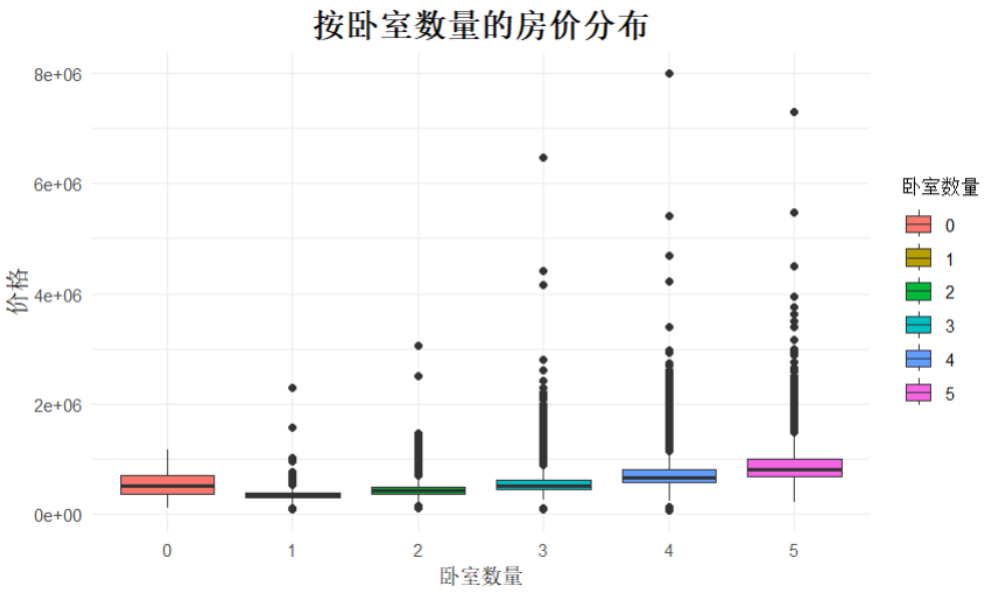
```
# 加载必要的库
library(tidyverse)    # 数据处理与可视化
library(lubridate)    # 日期处理
library(ggplot2)      # 绘图
library(ggthemes)     # 美化主题

# 读取数据
sales <- read.csv("D:/sales.csv")

# 转换日期格式
sales$datesold <- as.Date(sales$datesold, format = "%Y-%m-%d")

# 检查数据结构
str(sales)

# 按月汇总价格数据
monthly_data <- sales %>%
  mutate(month = floor_date(datesold, "month")) %>%
  group_by(month, propertyType) %>%
  summarise(mean_price = mean(price, na.rm = TRUE)) %>%
  ungroup()

# 绘制不同房产类型的月平均价格趋势
ggplot(monthly_data, aes(x = month, y = mean_price, color = propertyType)) +
```

```
  geom_line(size = 1) +
  labs(
    title = "不同房产类型月平均价格趋势",
    x = "月份",
    y = "平均价格",
    color = "房产类型"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, hjust = 0.5, face = "bold"),
    axis.title = element_text(size = 12),
    legend.title = element_text(size = 12)
  )

# 可视化邮政编码区域的房价分布
postcode_price <- sales %>%
  group_by(postcode) %>%
  summarise(mean_price = mean(price, na.rm = TRUE)) %>%
  arrange(desc(mean_price))

ggplot(postcode_price, aes(x = reorder(as.factor(postcode), mean_price), y =
mean_price)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  coord_flip() +
  labs(
    title = "不同邮政编码区域房价分布",
    x = "邮政编码",
    y = "平均价格"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, hjust = 0.5, face = "bold"),
    axis.title = element_text(size = 12)
  )

# 按卧室数量绘制价格分布箱线图
ggplot(sales, aes(x = as.factor(bedrooms), y = price, fill = as.factor(bedrooms))) +
  geom_boxplot() +
  labs(
    title = "按卧室数量的房价分布",
    x = "卧室数量",
    y = "价格",
    fill = "卧室数量"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, hjust = 0.5, face = "bold"),
    axis.title = element_text(size = 12)
  )
```

- 得到结果如下：







数据特性分析

- 对时间序列数据的平稳性和随机性进行检验，以评估数据是否适合直接建模。

1. **平稳性检验：**
   - 使用 ADF（Augmented Dickey-Fuller）检验判断时间序列是否平稳。
   - 若结果显示序列不平稳，通过对数变换或差分方法使其平稳。
1. **纯随机性检验：**
   - 使用 Ljung-Box 检验判断序列是否具有显著的自相关性。
   - 检验结果表明，房价序列存在显著的非随机性，适合建立时间序列模型。

- 相关代码如下：

```
# 加载必要的库
library(tseries)  # 包含 ADF 检验函数
library(forecast) # 包含差分与模型功能
library(stats)    # 包含 Ljung-Box 检验

# 读取数据并转为时间序列
sales <- read.csv("D:/sales.csv")
sales$datesold <- as.Date(sales$datesold, format = "%Y-%m-%d")

# 按月聚合房价数据（假设使用总房价时间序列）
monthly_price <- sales %>%
  mutate(month = floor_date(datesold, "month")) %>%
  group_by(month) %>%
  summarise(mean_price = mean(price, na.rm = TRUE)) %>%
  ungroup()

# 创建时间序列对象
price_ts <- ts(monthly_price$mean_price, start = c(2007, 1), frequency = 12) # 以2007
年为起点，频率为12（按月）

# 平稳性检验：ADF 检验
adf_result <- adf.test(price_ts, alternative = "stationary")
print("ADF 检验结果：")
print(adf_result)

# 如果数据不平稳，尝试对数变换和差分
if (adf_result$p.value > 0.05) {  # 检查 p 值是否大于 0.05（不平稳）
  print("数据不平稳，进行对数变换与差分处理")

  # 对数变换
  price_ts_log <- log(price_ts)

  # 差分（使平稳）
  price_ts_diff <- diff(price_ts_log)

  # 检查差分后序列的平稳性
  adf_diff_result <- adf.test(price_ts_diff, alternative = "stationary")
  print("对数变换 + 差分后 ADF 检验结果：")
  print(adf_diff_result)
```

```
  # 更新使用的时间序列
  price_ts <- price_ts_diff
} else {
  print("数据已平稳，无需变换或差分")
}

# 纯随机性检验：Ljung-Box 检验
ljung_box_result <- Box.test(price_ts, lag = 20, type = "Ljung-Box") # 设定滞后阶数为 20
print("Ljung-Box 检验结果：")
print(ljung_box_result)

if (ljung_box_result$p.value < 0.05) {
  print("时间序列存在显著的自相关性，非纯随机序列")
} else {
  print("时间序列是纯随机序列")
}
```

1. **平稳性检验：**

- ADF检验结果输出如下图：

```
[1] "ADF 检验结果："
> print(adf_result)

        Augmented Dickey-Fuller Test

data:  price_ts
Dickey-Fuller = -2.862, Lag order = 5, p-value = 0.2173
alternative hypothesis: stationary
```

- 因为 ADF 检验的 p-value 大于 0.05，因此无法拒绝原假设，表示该时间序列非平稳。
  原始数据存在趋势或波动性，不适合直接进行建模，所以进一步进行预处理（对数变换和差分），进行对数变换和差分处理后再次进行ADF检验输出结果如下图：

```
[1] "数据不平稳，进行对数变换与差分处理"
[1] "对数变换 + 差分后 ADF 检验结果："

        Augmented Dickey-Fuller Test

data:  price_ts_diff
Dickey-Fuller = -8.3537, Lag order = 5, p-value = 0.01
alternative hypothesis: stationary
```

- 差分后的数据通过 ADF 检验，p-value 小于 0.05，拒绝原假设，证明时间序列已平稳。

2. **纯随机性检验：**

- Ljung-Box 检验（纯随机性检验）结果输出如下图：

```
[1] "Ljung-Box 检验结果："
> print(ljung_box_result)

        Box-Ljung test

data:  price_ts
X-squared = 76.29, df = 20, p-value = 1.658e-08
```

- p-value 小于 0.05，表示时间序列存在显著的自相关性，时间序列非纯随机序列，即它的当前值与过去的数据存在相关性，可以进一步用自回归（AR）等模型进行建模。

数据的初步观察结论

- 房价的时间序列表现出显著的季节性和趋势性特征，尤其在 House 类型中较为明显。
- 房价受区域（邮政编码）和卧室数量的显著影响，不同房产类型的价格变化模式存在差异。
- 数据基本适合通过时间序列模型进行建模，但需要进一步的预处理以优化模型输入。

3. 数据建模与模型质量评估

- 在本部分，我们基于给定的房产销售数据，采用时序分析方法，选择最优的ARIMA模型来对不同类型房产的销售价格进行预测。以下是详细的步骤和实现过程：

数据预处理

- 首先，我们对原始房产销售数据进行了预处理。具体步骤如下：
- **加载数据**：读取存储在CSV文件中的房产销售数据，并将其转换为适用于分析的格式。

```
sales_data <- read.csv("sales.csv", stringsAsFactors = FALSE)
```

- **处理缺失值**：使用na.omit函数移除含有缺失值的行，确保后续的分析不会受到缺失数据的影响。

```
sales_data <- na.omit(sales_data)
```

- **日期处理**：将datesold字段转换为日期格式，并提取出销售的月份和年份信息，为后续的时间序列建模提供基础。

```
sales_data$datesold <- as.Date(sales_data$datesold)
sales_data <- sales_data %>%
  mutate(month = floor_date(datesold, "month"),
         year = year(datesold))
```

- **计算每月平均价格**：按每个月、每种房产类型计算平均销售价格，得到monthly_avg_price数据集。这个数据集将作为时序模型的输入。

```
monthly_avg_price <- sales_data %>%
  group_by(year, month, propertyType) %>%
  summarise(avg_price = mean(price, na.rm = TRUE)) %>%
  ungroup()
```

时序模型建模

- 在数据预处理后，我们构建了时序预测模型。具体过程如下：
  - **创建时间序列对象**：对每种房产类型，使用avg_price计算每月的房产销售价格时间序列。

```
ts_data <- ts(df$avg_price, start = c(min(df$year), min(format(df$month, "%m"))),
frequency = 12)
```

  - **分割训练集和测试集**：为了验证模型的预测效果，我们将数据集分为80%的训练集和20%的测试集。

```
train_size <- floor(0.8 * length(ts_data))
train_data <- window(ts_data, end = c(time(ts_data)[train_size], 12))
test_data <- window(ts_data, start = c(time(ts_data)[train_size + 1], 12))
```

- **选择ARIMA模型**：使用auto.arima函数自动选择最优的ARIMA模型。这个过程通过自动选择差分阶数（d和D）以及是否包含季节性（seasonal）来构建最适合数据的模型。

```
fit <- auto.arima(train_data, d = 1, D = 1, seasonal = TRUE)
```

- **预测与误差评估**：在建立好ARIMA模型后，我们使用模型对测试集进行预测，并计算模型的准确度指标，例如误差率、均方根误差等。

```
pred <- forecast(fit, h = length(test_data))
accuracy_metrics <- accuracy(pred, test_data)
```

- **未来价格的预测**：使用拟合的ARIMA模型进行未来价格的预测。预测结果包括未来12个月的房产销售价格。

```
future_forecast <- forecast(fit, h = 12)
```

- **结果保存**：将预测的结果转换为数据框，并存储为CSV文件，方便后续查看。

```
future_df <- data.frame(
    date = seq(max(df$month) + months(1), by = "month", length.out = 12),
    price = future_forecast$mean,
    propertyType = unique(df$propertyType)
)
```

模型评估与结果输出

- 评估模型的准确度：对于每个房产类型，我们输出模型的准确度，包括预测误差（如RMSE、MAE等），并保存到文件中。

```
write.csv(result$accuracy, paste(current_property_type, "model_accuracy.csv",
collapse=''), row.names = FALSE)
```

- 得到模型准确性分析结果如下：

（1）house模型性能

	A	B	C	D	E	F	G	H	I
1		ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
2	Test set	-1430.8	44074.6	29744.1	-0.4892	5.14708	0.65714	0.06474	NA
3	Training set	-31164	55204	43399.3	-4.4557	6.0564	0.95882	0.17167	1.38421

  - **误差（ME）**：预测值整体略低于实际值（负值），可能反映模型未能完全捕捉高价格趋势。
  - **均方根误差（RMSE）和平均绝对误差（MAE）**：存在误差值，可能是由于某些月份价格波动剧烈。
  - **MAPE**：相对误差平均在 5.6%，属于中等预测精度范围。

- **自相关（ACF1）**：误差有轻微的正自相关，可能有改进空间。

（2）unit模型性能

	A	B	C	D	E	F	G	H	I
1		ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
2	Test set	-1987.7	45293.3	30167	-1.0412	7.04731	0.69303	-0.0546	NA
3	Training set	-3244.8	19278.9	14670.5	-0.9644	3.50851	0.33703	-0.3566	0.62245

- **误差（ME）**：预测值比实际值稍低，但误差幅度较小，模型稳定性较好。
- **均方根误差（RMSE）和平均绝对误差（MAE）**：单位模型表现优于房屋模型，误差相对较低。
- **MAPE**：单位模型的相对误差为 5.28%，精度稍高于房屋模型。
- **自相关（ACF1）**：负相关值表明预测误差之间的独立性较强。

- 保存预测结果：所有预测结果将被保存到predicted_prices.csv文件中，便于查看和进一步分析。

```
write.csv(predictions, "predicted_prices.csv", row.names = FALSE)
```

- 输出predicted_prices.csv预测值如图：

	A	B	C	D
1	date	price	propertyType	
2	2019/8/1	681581.6469	house	
3	2019/9/1	750997.1025	house	
4	2019/10/1	754155.1078	house	
5	2019/11/1	733374.8271	house	
6	2019/12/1	765650.1435	house	
7	2020/1/1	729584.4214	house	
8	2020/2/1	735623.8785	house	
9	2020/3/1	743007.0146	house	
10	2020/4/1	767661.8296	house	
11	2020/5/1	771047.745	house	
12	2020/6/1	830087.8634	house	
13	2020/7/1	777062.1493	house	
14	2019/8/1	414162.1032	unit	
15	2019/9/1	423172.0584	unit	
16	2019/10/1	424347.5217	unit	
17	2019/11/1	431808.1893	unit	
18	2019/12/1	426533.2463	unit	
19	2020/1/1	410520.2635	unit	
20	2020/2/1	408131.9879	unit	
21	2020/3/1	418735.2476	unit	
22	2020/4/1	430849.4313	unit	
23	2020/5/1	434897.3736	unit	
24	2020/6/1	423905.4407	unit	
25	2020/7/1	428888.7026	unit	

- 具体预测结果如下：

  house未来12个月预测值

date	price	propertyType
2019/8/1	681581.6469	house
2019/9/1	750997.1025	house
2019/10/1	754155.1078	house
2019/11/1	733374.8271	house
2019/12/1	765650.1435	house
2020/1/1	729584.4214	house
2020/2/1	735623.8785	house
2020/3/1	743007.0146	house
2020/4/1	767661.8296	house
2020/5/1	771047.7450	house
2020/6/1	830087.8634	house
2020/7/1	777062.1493	house

unit未来12个月预测值

date	price	propertyType
2019/8/1	414162.1032	unit
2019/9/1	423172.0584	unit
2019/10/1	424347.5217	unit
2019/11/1	431808.1893	unit
2019/12/1	426533.2463	unit
2020/1/1	410520.2635	unit
2020/2/1	408131.9879	unit
2020/3/1	418735.2476	unit
2020/4/1	430849.4313	unit
2020/5/1	434897.3736	unit
2020/6/1	423905.4407	unit
2020/7/1	428888.7026	unit

- 数据建模部分相关完整代码如下：

```r
# 加载必要的库
library(dplyr)
library(lubridate)
library(ggplot2)
library(forecast)

# 读取CSV文件
setwd("D:/homework/r语言")
sales_data <- read.csv("sales.csv", stringsAsFactors = FALSE)

# 1. 数据预处理

# 检查是否有缺失值并移除含有缺失值的行
sales_data <- na.omit(sales_data)

# 转换datesold为日期格式
sales_data$datesold <- as.Date(sales_data$datesold)

# 添加月份和年份列
sales_data <- sales_data %>%
  mutate(month = floor_date(datesold, "month"),
         year = year(datesold))

# 计算每个月每个房产类型的平均价格
monthly_avg_price <- sales_data %>%
  group_by(year, month, propertyType) %>%
  summarise(avg_price = mean(price, na.rm = TRUE)) %>%
  ungroup()

# 2. 使用最优的时间序列模型建模
# 创建一个函数来处理每个组合的时间序列
forecast_model <- function(df) {
  # 创建时间序列对象
  ts_data <- ts(df$avg_price, start = c(min(df$year), min(format(df$month, "%m"))),
frequency = 12)
  #print(ts_data)
  # 分离训练集和测试集
  train_size <- floor(0.8 * length(ts_data))
  train_data <- window(ts_data, end = c(time(ts_data)[train_size], 12))
  test_data <- window(ts_data, start = c(time(ts_data)[train_size + 1], 12))

  print(train_data)
  # 自动选择最优ARIMA模型
  fit <- auto.arima(train_data, d = 1, D = 1, seasonal = TRUE)

  # 预测测试集
  pred <- forecast(fit, h = length(test_data))

  # 计算误差
  accuracy_metrics <- accuracy(pred, test_data)

  # 预测未来12个月
```

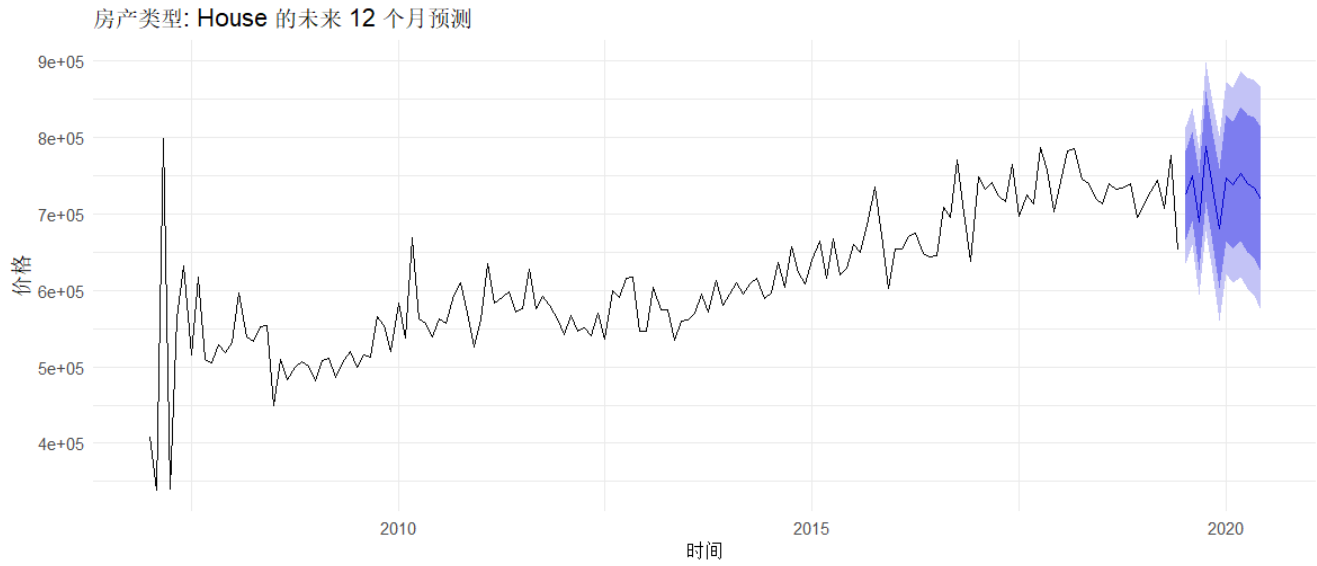
```
  future_forecast <- forecast(fit, h = 12)
  print(future_forecast)
  # 提取预测结果并转换为数据框
  future_df <- data.frame(
    date = seq(max(df$month) + months(1), by = "month", length.out = 12),
    price = future_forecast$mean,
    propertyType = unique(df$propertyType)
  )
  return(list(
    accuracy = accuracy_metrics,
    future_forecast = future_df
  ))
}

# 对每个房产类型应用时间序列建模，并收集评估结果
predictions <- monthly_avg_price %>%
  group_by(propertyType) %>%
  do({
      result <- forecast_model(.)
  # 3.打印保存评估结果
  current_property_type <- unique(.$propertyType)
  print(current_property_type)
  print(result$accuracy)

 write.csv(result$accuracy,paste(current_property_type,"model_accuracy.csv",collapse=
''), row.names = FALSE)
      result$future_forecast
  }) %>%
  ungroup()
# 4. 将预测结果保存到新的CSV文件
write.csv(predictions, "predicted_prices.csv", row.names = FALSE)
```

可视化预测结果

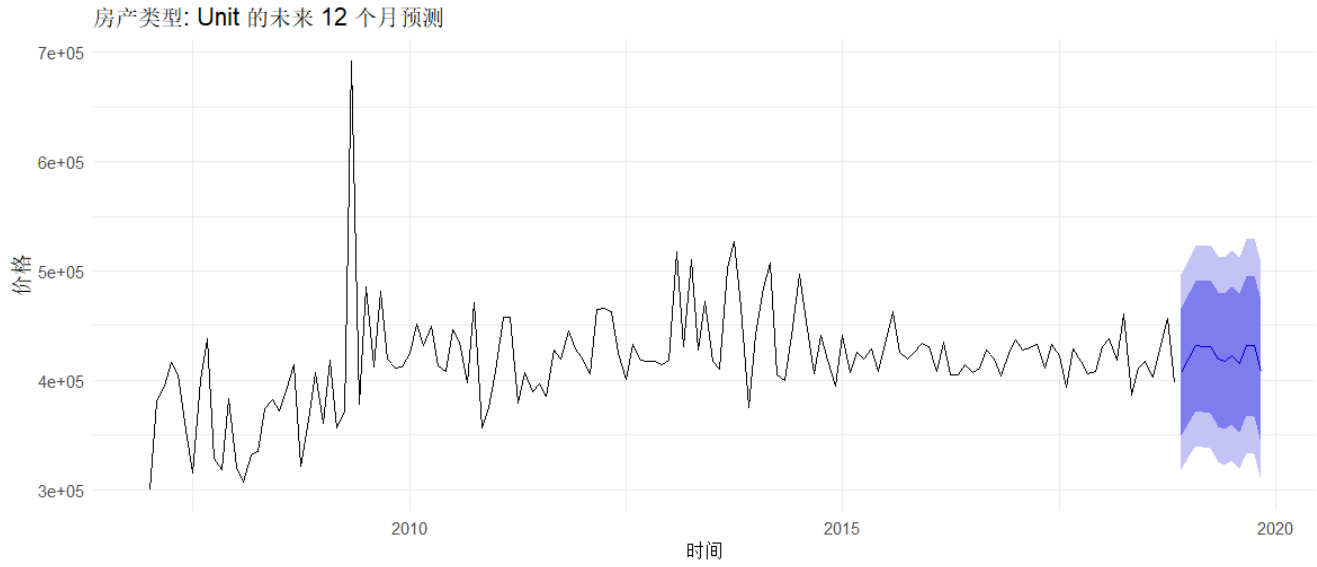
- 为了更直观地展示预测结果，我们使用`ggplot2`进行预测数据的可视化。通过绘制未来12个月预测的房产价格趋势图，帮助房产中介或购房者做出决策。

```
autoplot(house_forecast) +
  ggtitle("房产类型：House 的未来 12 个月预测") +
  xlab("时间") +
  ylab("价格")

autoplot(unit_forecast) +
  ggtitle("房产类型：Unit 的未来 12 个月预测") +
  xlab("时间") +
  ylab("价格")
```

house未来12个月预测值可视化



unit未来12个月预测值可视化



4. 模型解读与建议

基于对时间序列数据的建模与预测结果分析，本部分对模型预测的房价趋势进行解读，并针对房地产中介和购房者提供实际建议，以支持市场决策。

模型预测结果解读

1. **整体趋势分析：**

- **House 类型房产**：未来 12 个月价格呈现一定的波动性，初期价格较高，随后略有下降并趋于稳定。例如，预测显示2019 年 11 月低至 729584.4 元,而 2020 年 6 月房价上升达到 830087.9元。这可能反映市场的季节性变化或需求波动。
- **Unit 类型房产**：价格基本保持稳定，预测均值在408132.9元到434897.4元之间。置信区间较窄，表明市场对该类型房产的价格波动较小，具有较高的稳定性。

2. **房产类型差异：**

- **House 类型：** 价格波动幅度较大，说明其受市场需求影响显著，可能与区域热点和政策调整有关。

- **Unit 类型：** 价格波动较小，表现出更高的市场稳定性，适合风险规避型买家。

对房地产中介及购房者的建议

- 根据预测结果，未来12个月中，不同类型房产的价格走势显示出显著的特点，为房产中介和购房者提供了重要的市场洞察。对于独立住宅（house），价格整体呈现出一定的波动性。从2019年8月的681,581元开始，价格在随后几个月逐步上升，到2019年10月达到了754,155元的高点。虽然此后价格在短期内略有回落，但总体趋势仍维持在较高水平，并在2020年4月达到767,661元的峰值。这种价格波动可能反映了市场对独立住宅持续强劲的需求，同时也提示购房者应在价格回落阶段积极关注市场，以获取更具性价比的选择。
- 相比之下，公寓（unit）的价格趋势更加稳定，波动幅度较小。预测显示，公寓价格从2019年8月的414,162元开始，逐步小幅上涨，在2020年4月达到430,849元。这一趋势表明公寓市场具有较强的稳定性，可能受到投资者和首次购房者的青睐。尤其是在市场动荡或整体经济环境存在不确定性时，公寓因其更低的总价和较高的租赁回报率而成为稳健的投资选择。
- 从两类房产的价格预测来看，独立住宅和公寓的市场表现各有侧重。对于追求长期增值的购房者，独立住宅的潜在回报更为可观，但需要注意价格的短期波动风险。而公寓的价格则因稳定性和相对较低的入市门槛，适合预算有限或寻求稳定收益的买家。这一预测还为房产中介提供了销售策略调整的依据，例如在市场波动期加强对独立住宅的推广，而在稳定期突出公寓的投资价值。
- 综合来看，无论是购房者还是投资者，都可以根据自身需求和风险承受能力，结合预测数据制定更科学的决策。未来的市场表现可能会受到政策、经济环境等外部因素的影响，因此建议密切关注市场动态，并在合适的时间抓住投资机会。

优化市场决策的潜在方向

1. **结合外部数据进行分析：**

- 考虑加入影响房价的外部变量（如利率、政策调整）进行分析，以提高模型解释力。
- 对历史节假日和政策实施日期进行回归分析，优化季节性预测。

2. **动态更新预测模型：**

- 建立实时数据采集和预测机制，动态更新模型，以适应快速变化的市场环境。
- 通过定期更新数据，捕捉市场需求的最新变化趋势。

3. **开发用户友好型工具：**

- 为中介开发可视化工具，展示房价预测结果，提升数据驱动决策能力。
- 为购房者提供互动式平台，模拟不同场景下的房价变化，为购房决策提供参考。

5. 模型不足与优化建议

尽管本项目通过 ARIMA 模型对房价进行了较为准确的预测，但仍存在一些局限性，需要在后续工作中优化以提升预测能力和解释性。

模型不足

- 尽管本项目采用了ARIMA模型对房价进行了预测，并取得了较为理想的效果，但在实际应用中仍暴露出一些不足之处。首先，现有模型对季节性特征的处理较为简单，未能充分捕捉复杂的周期规律。例如，某些房产类型的价格高峰和低谷并未得到深入解析。此外，节假日等特殊时间节点可能显著影响购房需求，但未被纳入模型，这限制了模型对市场动态的全面捕捉。
- 其次，模型过于依赖历史数据，未能引入利率、收入水平、购房政策等外部变量，这导致模型的解释能力不足。经济波动和政策调整对房价的影响未能体现在模型中，这使得预测结果难以全面反映市场变动。对于房产中介和购房者来说，这一不足可能降低模型的实用性。
- 再者，ARIMA模型在短期预测中表现较好，但其长期预测能力相对较弱，特别是在市场动态变化较大的情况下。模型假设市场趋势和季节性特征在预测期内保持不变，但实际情况中市场常常受到政策或非线性因素（如购房者预期等）的影响。此外，部分房产类型的残差未完全为白噪声，表明模型尚未充分拟合所有时间序列信息。

优化建议

1. **引入外部变量：**

- 为了进一步提升模型的预测能力，我们建议引入更多外部变量以增强对房价波动的理解和解释能力。宏观经济指标如利率、就业率和地区GDP增长率等，是影响房价的重要因素，将其纳入模型能够更全面地反映外部驱动因素。此外，政策因素也不容忽视，例如历史购房补贴政策或限购政策的变化可能显著影响房价走势。将节假日节点数据整合进模型，也有助于捕捉季节性需求的变化。

2. **分层建模：**

- 在建模过程中，分层分析是一个值得尝试的方向。对不同区域的房价单独建模，能够提升区域层面的适应性，避免“一刀切”的泛化问题。结合空间数据进行分析，例如利用房产热力图或地理信息系统工具，可以更直观地呈现不同地段的房价分布特点，从而优化模型的空间预测能力。

3. **尝试复杂模型：**

- 可以尝试使用更复杂的预测模型来提升性能。Prophet 模型因其易于捕捉时间序列趋势和季节性特征而备受推荐，特别适合结合外部变量进行分析。对于非线性时间序列，LSTM（长短期记忆网络）是一种强有力的工具，能够捕捉复杂的动态变化。而 SARIMA 和 GARCH 模型则在处理季节性特征和时间序列波动性方面表现卓越，是传统模型的优秀补充。

4. **增强残差诊断与修正：**

- 优化残差的诊断与修正同样至关重要。未被解释的残差动态可能隐藏了更多的信息。可以通过调整模型阶数或引入混合模型（如 SARIMA 和 GARCH 的结合）来进一步优化。如果 Ljung-Box 检验结果表明模型残差未通过白噪声检验，则需重新调整参数，确保模型的准确性。

5. **动态更新与实时预测：**

- 动态更新和实时预测是应对快速变化市场需求的关键策略。采用滚动预测机制，利用最新数据动态更新模型，不仅能提高预测的实时性，还能在市场环境变化时快速响应。开发自动化框架以简化模型的选择、评估和优化流程，也将大幅提高分析效率。

6. **用户友好型报告与工具：**

- 提升预测结果的用户友好性也非常重要。通过交互式数据可视化工具，例如基于 Shiny 的应用程序，为房产中介和购房者提供直观的房价预测报告，支持其决策。多场景模拟功能，例如在不同利率假设下的房价走势预测，能帮助用户更好地理解和应对潜在风险。