**机器学习大作业**

|  |  |
| --- | --- |
| **作业题目：** | **大数据线性回归预测学生成绩** |
| **任课老师：**  **学生姓名：** |  |
|  |
| **学 院 ：** | **网络安全与信息技术学院** |
| **班 级 ：** | **计科21-2** |
| **学 号：** | **211181510** |
| **时 间：** | **2024年6月24日** |

[**1　概述** 1](#_Toc170335335)

[**2　实验环境** 1](#_Toc170335336)

[**3　实验数据及分析** 1](#_Toc170335337)

[**3.1实验数据背景及位置** 1](#_Toc170335338)

[**3.2 数据集格式** 1](#_Toc170335339)

[**3.3 实验分析** 2](#_Toc170335340)

[**4　实验涉及知识点** 2](#_Toc170335341)

[**5　实验过程** 3](#_Toc170335342)

[**5.1 导入库** 3](#_Toc170335343)

[**5.2 导入并初始化数据** 3](#_Toc170335344)

[**5.3 数据处理** 4](#_Toc170335345)

[**6.实验结论** 14](#_Toc170335346)

[**7 附录** 15](#_Toc170335347)

[**7.1实验代码** 15](#_Toc170335348)

[**7.2数据集的属性介绍** 20](#_Toc170335349)

**1　概述**

线性回归是一种统计学方法，用于建立一个或多个自变量（解释变量）与因变量（响应变量）之间的线性关系。它是一种预测分析技术，用于预测连续的数值型数据。

现需使用此方法通过训练集训练和测试集测试学校学生的信息属性来生成多个[线性模型](https://so.csdn.net/so/search?q=%E7%BA%BF%E6%80%A7%E6%A8%A1%E5%9E%8B&spm=1001.2101.3001.7020)，预测学生成绩。

**2　实验环境**

Windows 11操作系统

Pycharm 2023

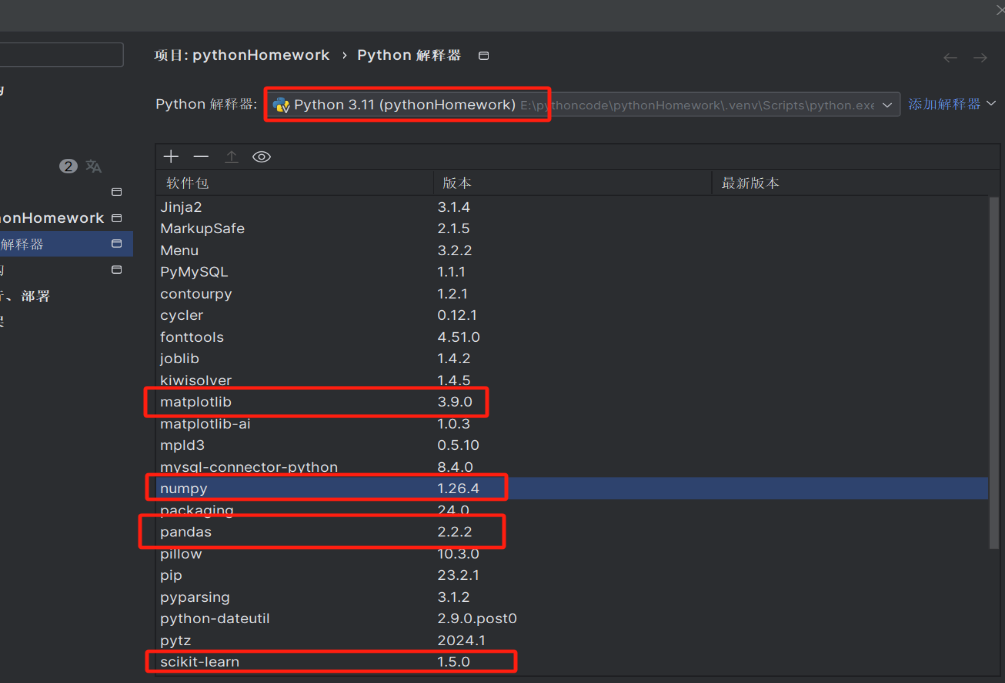
Python 3.11

numpy 1.26.4

pandas 2.2.2

scikit-learn 1.5.0

matplotlib 3.9.0



**3　实验数据及分析**

**3.1实验数据背景及位置**

现在有一份数据集，该数据接近了学校学生的学习成绩。数据属性包括学生成绩，人口[统计学](https://so.csdn.net/so/search?q=%E7%BB%9F%E8%AE%A1%E5%AD%A6&spm=1001.2101.3001.7020)，社会和与学校相关的特征，并通过使用学校报告和调查表进行收集。提供了两个关于两个不同学科表现的数据集：数学（mat）和葡萄牙语（por）

数据集存放在"C:\Users\裕龙\Desktop\机器学习\线性回归预测成绩r\BigDataPredicteGrades-master\student-mat.csv"

**3.2 数据集格式**

该数据集共有396条，每列33个属性。

33个属性分别是：学校、性别、年龄、地址-、famsize、Pstatus、Medu、Fedu、Mjob、Fjob、理由、监护人、traveltime、学习时间、失败、schoolup、famsup、付费、、托儿所、更高、互联网-、浪漫、家族、空闲时间、外出、Dalc、Walc、健康、缺勤、G1、G2、G3。如图1学生信息数据集所示：

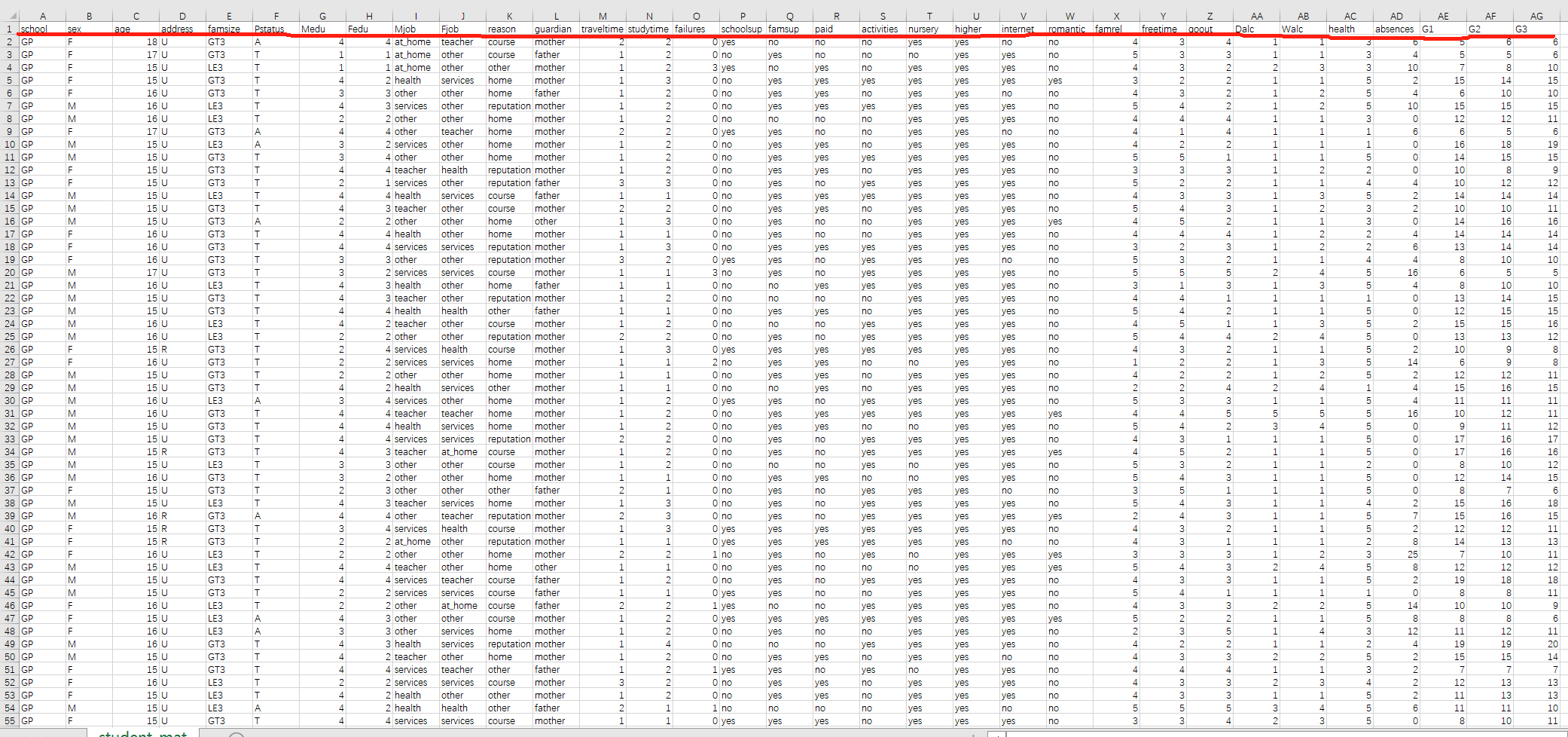


图1 学生信息数据集

**3.3 实验分析**

·使用python语言，导入相应的库，导入所需的数据，做数据的预处理

·构建简单线性回归模型

·使用数据集中不同的参数，进行线性回归模型的比较

**4　实验涉及知识点**

线性回归模型

**定义：**

线性回归尝试找到一个线性方程，该方程能够最好地描述自变量和因变量之间的关系。

**类型：**

* **简单线性回归**：只涉及一个自变量和一个因变量。
* **多元线性回归**：涉及两个或更多自变量。

**形式：**

线性回归模型通常表示为： y=β0+β1x1+β2x2+...+βnxn+ϵ*y*=*β*0​+*β*1​*x*1​+*β*2​*x*2​+...+*βn*​*xn*​+*ϵ* 其中：

* y*y* 是因变量。
* x1,x2,...,xn*x*1​,*x*2​,...,*xn*​ 是自变量。
* β0*β*0​ 是截距项。
* β1,β2,...,βn*β*1​,*β*2​,...,*βn*​ 是各自变量的系数。
* ϵ*ϵ* 是误差项，代表模型无法解释的随机变异。

**目标：**

* 确定自变量与因变量之间的线性关系。
* 预测因变量的值。
* 分析自变量对因变量的影响。

**方法：**

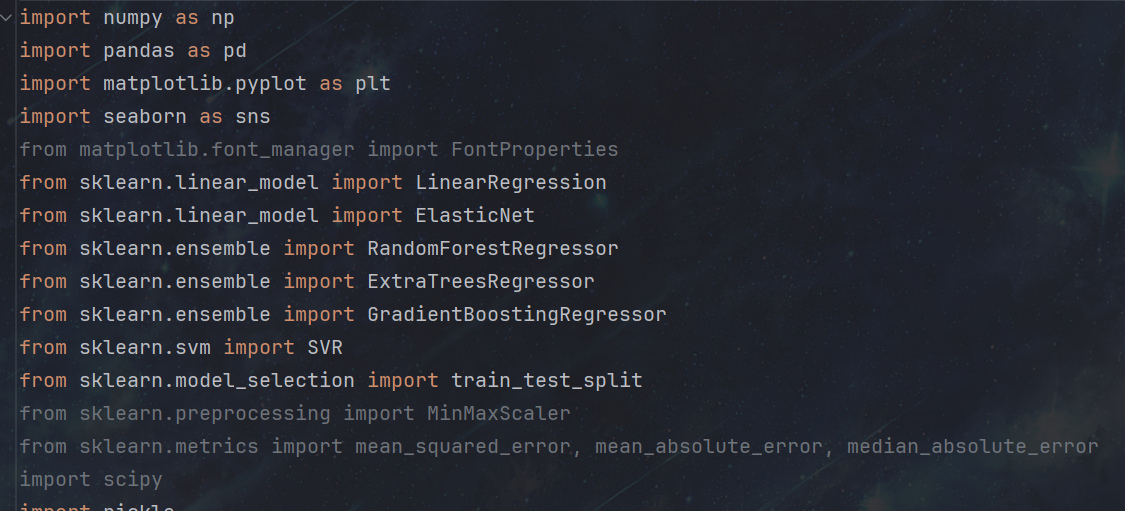
* **最小二乘法**：通过最小化实际观测值和模型预测值之间的平方差来估计模型参数。

**评估：**

* **均方误差（MSE）**：衡量预测误差的平方的平均值。
* **均方根误差（RMSE）**：MSE的平方根，提供了与原始数据相同单位的误差度量。
* **决定系数（R²）**：衡量模型解释的变异性的比例。

**5　实验过程**

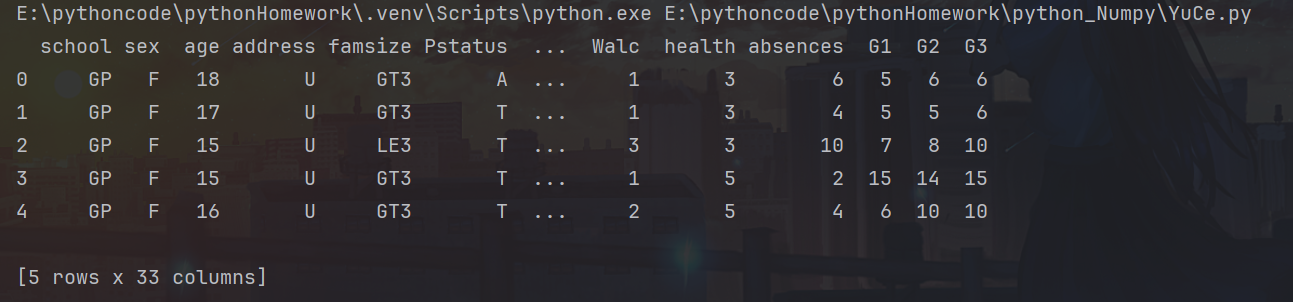
**5.1 导入库**



**5.2 导入并初始化数据**

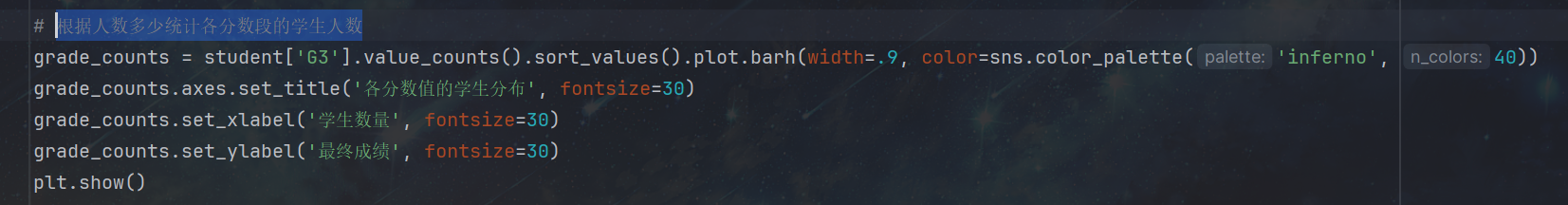


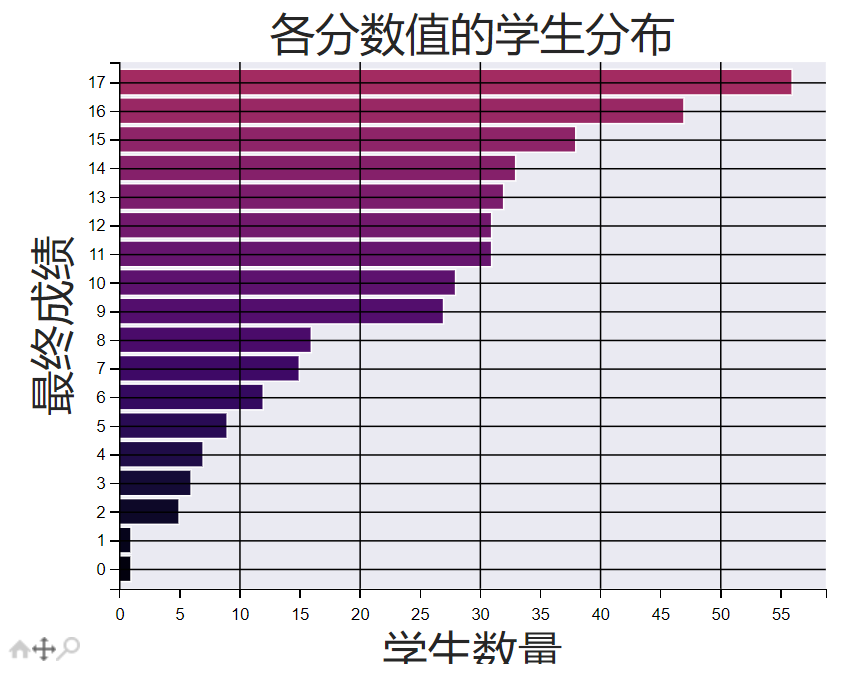
输出结果：



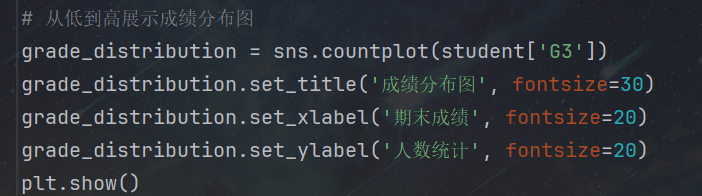
**5.3 数据处理**

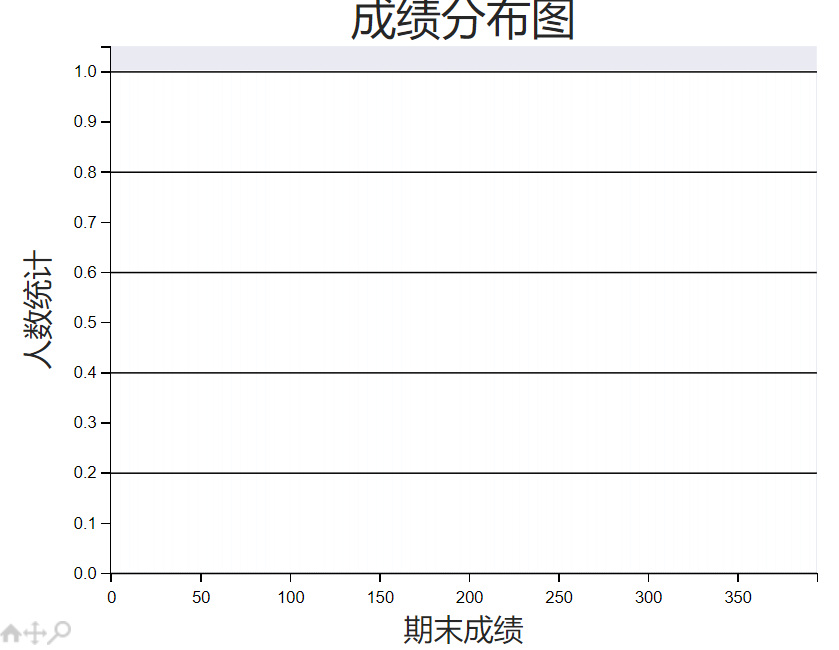
5.3.1 根据人数多少统计各分数段的学生人数



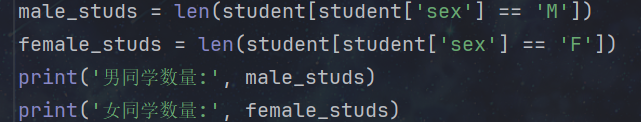


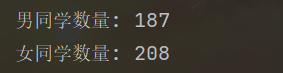
5.3.2 从低到高展示成绩分布图





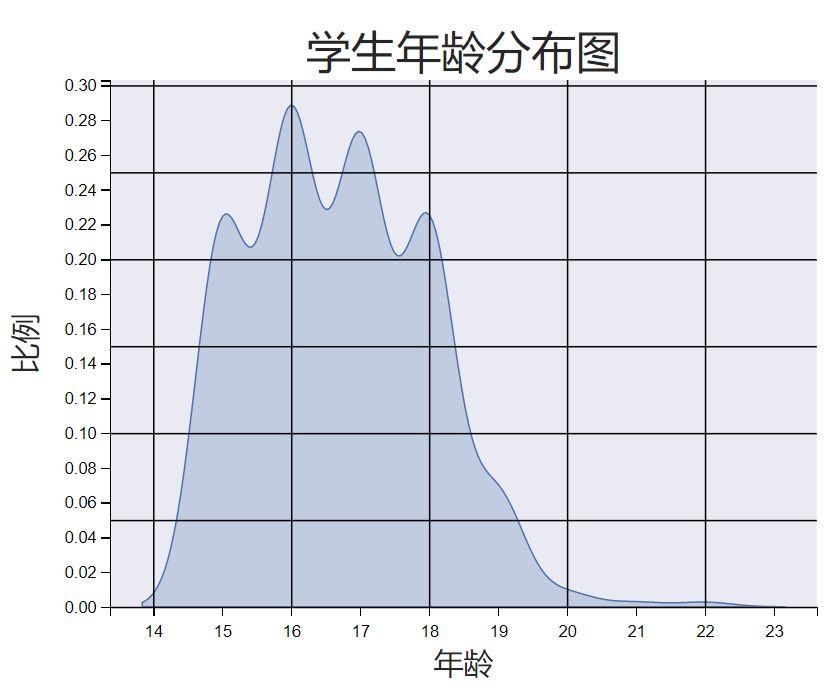
5.3.4分析性别比例



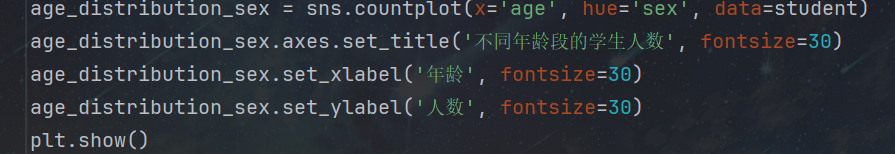


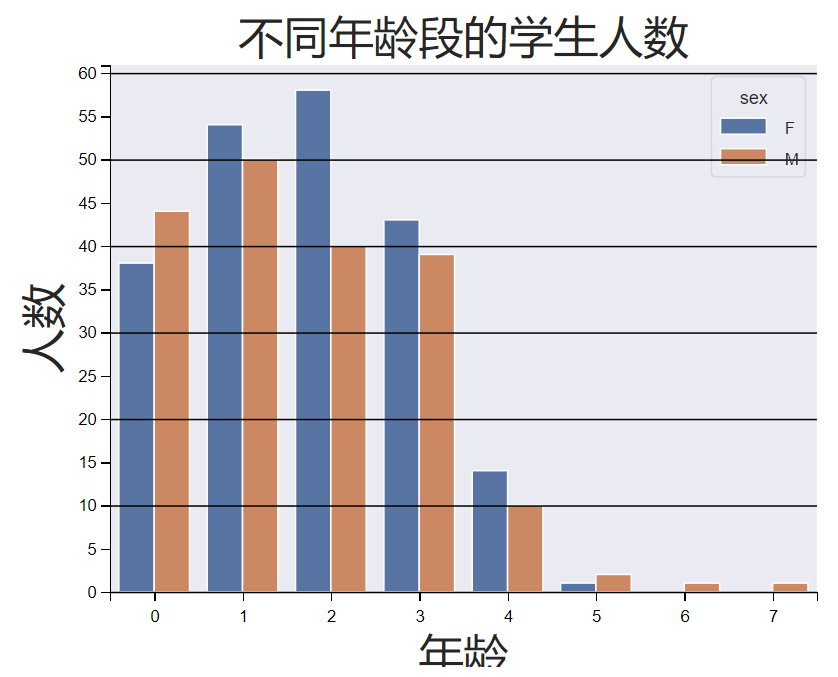
5.3.5 分析年龄分布比例（曲线图）



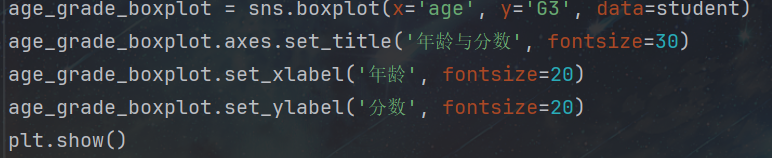


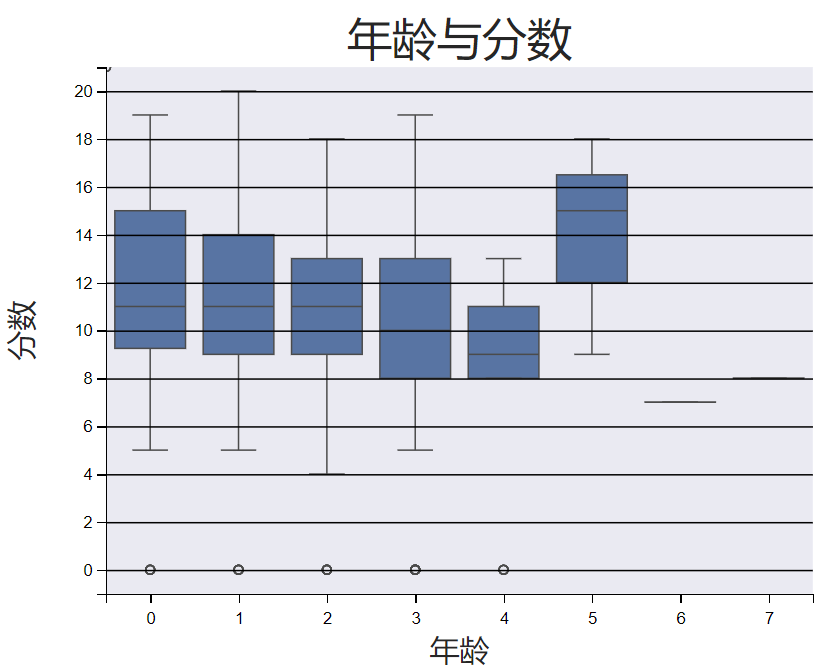
5.3.6 分性别年龄分布图（柱状图）



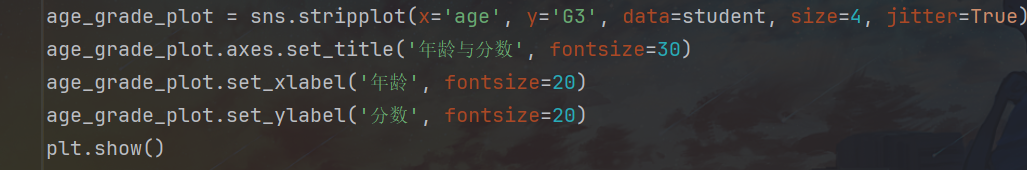


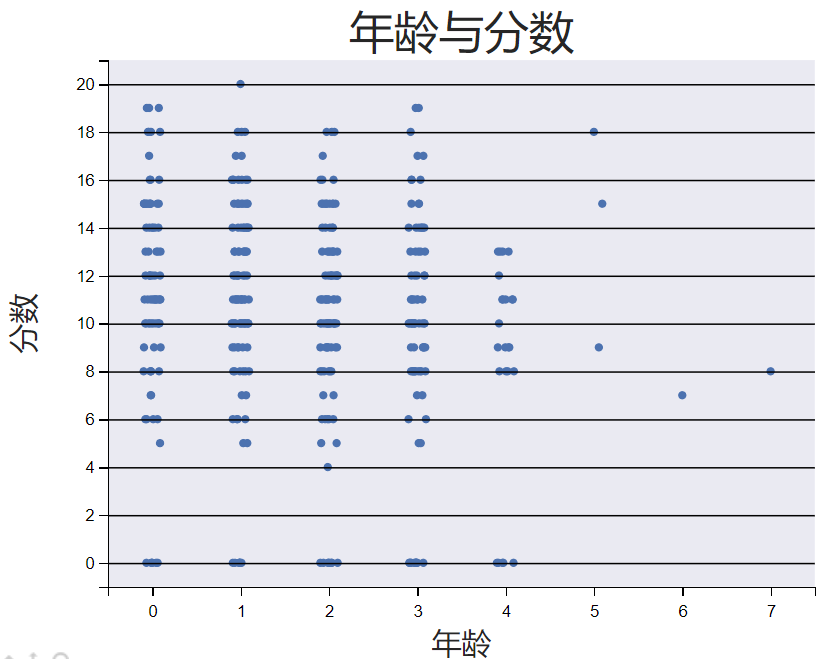
5.3.7 各年龄段的成绩箱型图



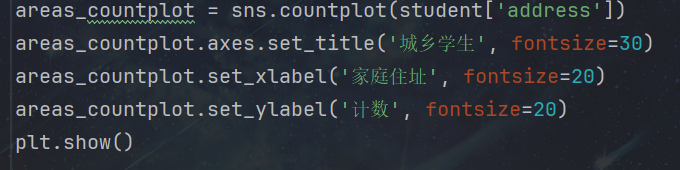


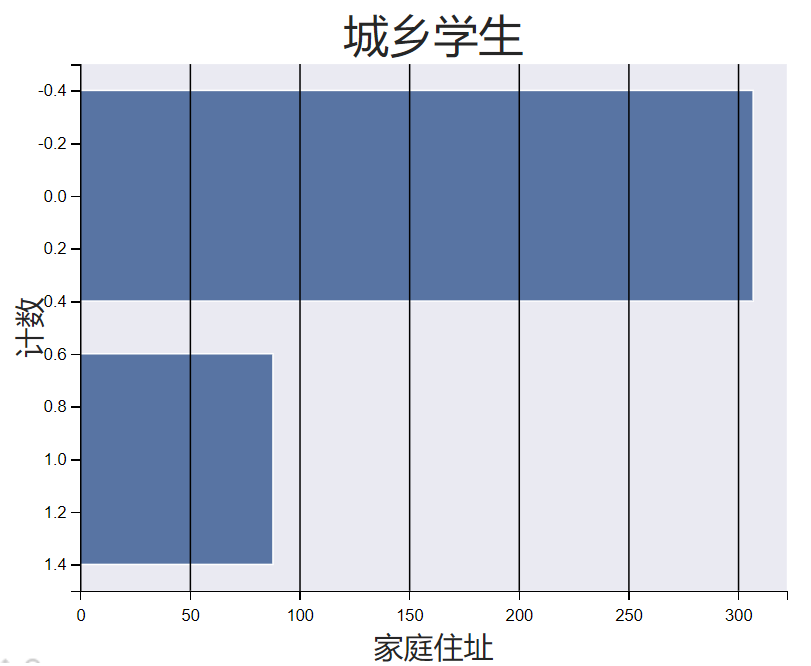
5.3.8 各年龄段的成绩分布图





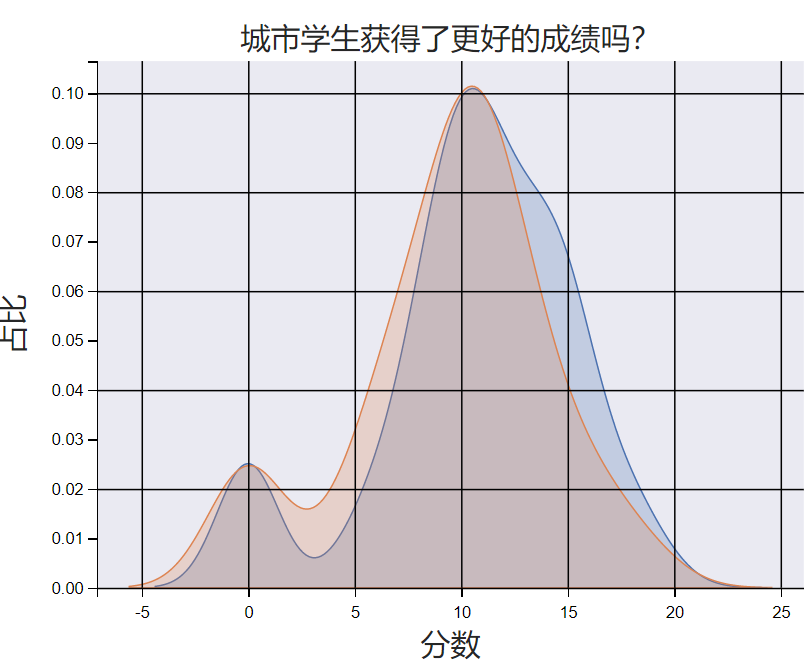
5.3.9 城乡学生计数



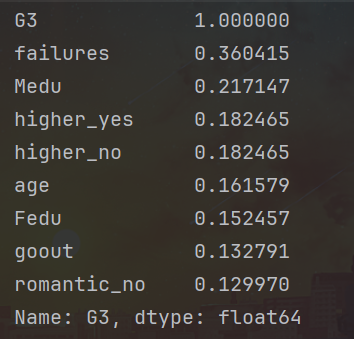


5.3.10 城市学生获得了更好的成绩吗

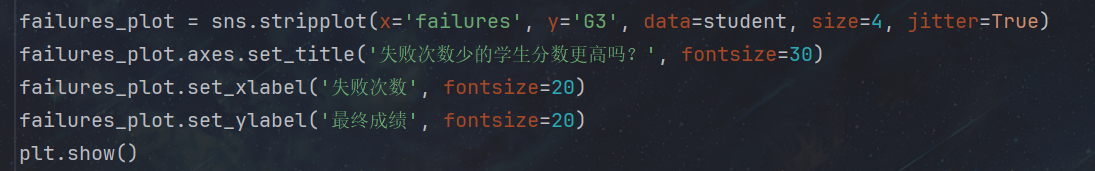


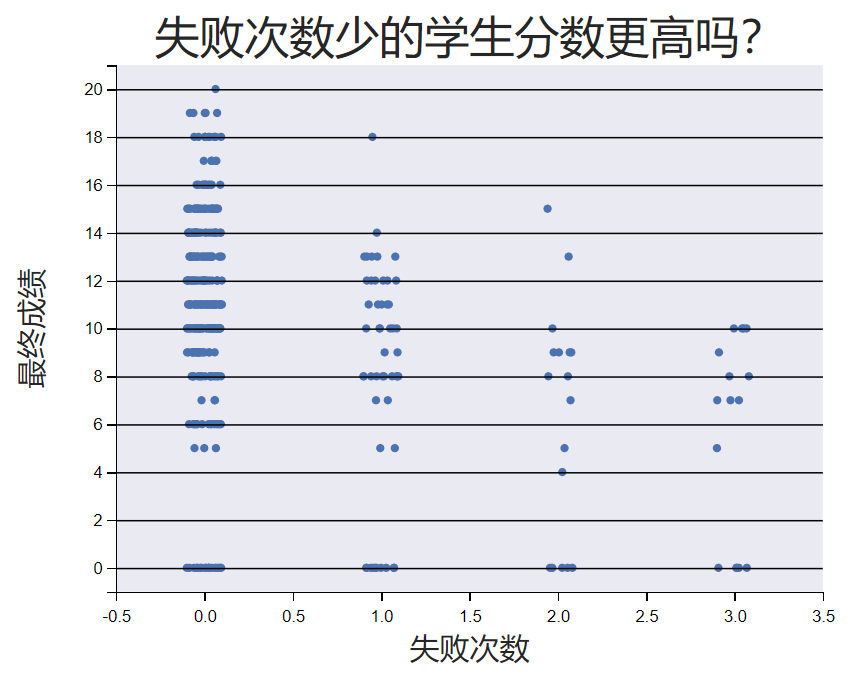


5.3.11 选取G3属性值，选取相关性最强的8个

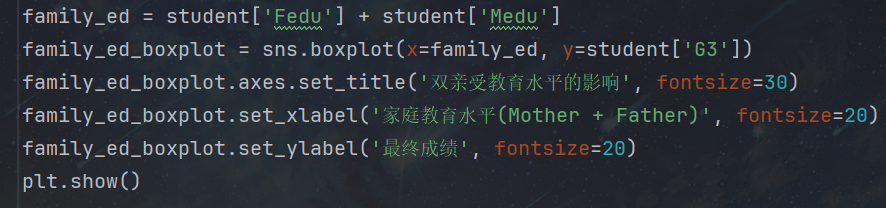
 

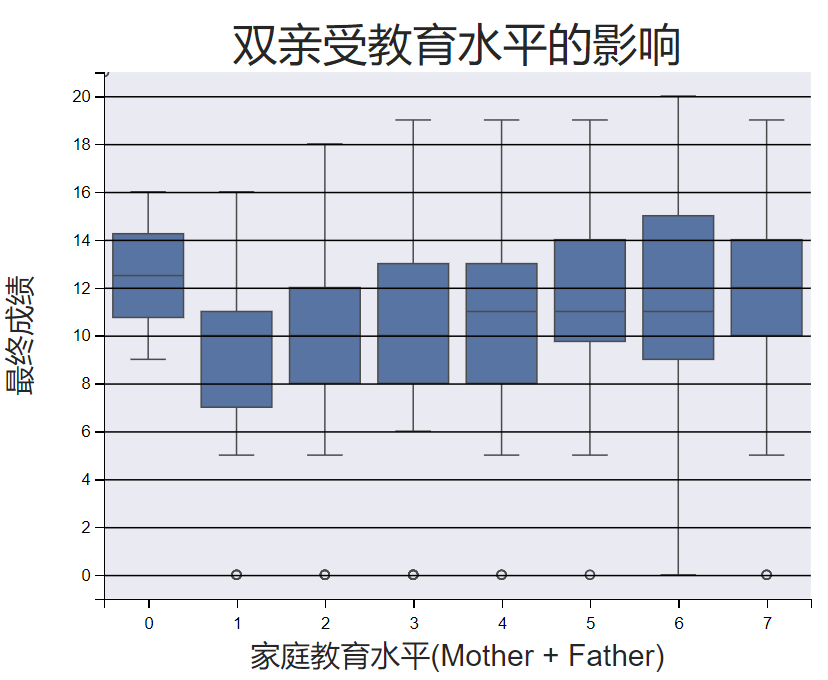
5.3.12 失败次数成绩分布图



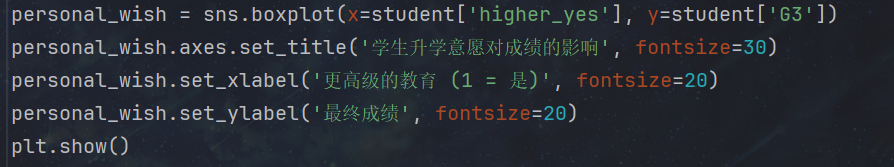


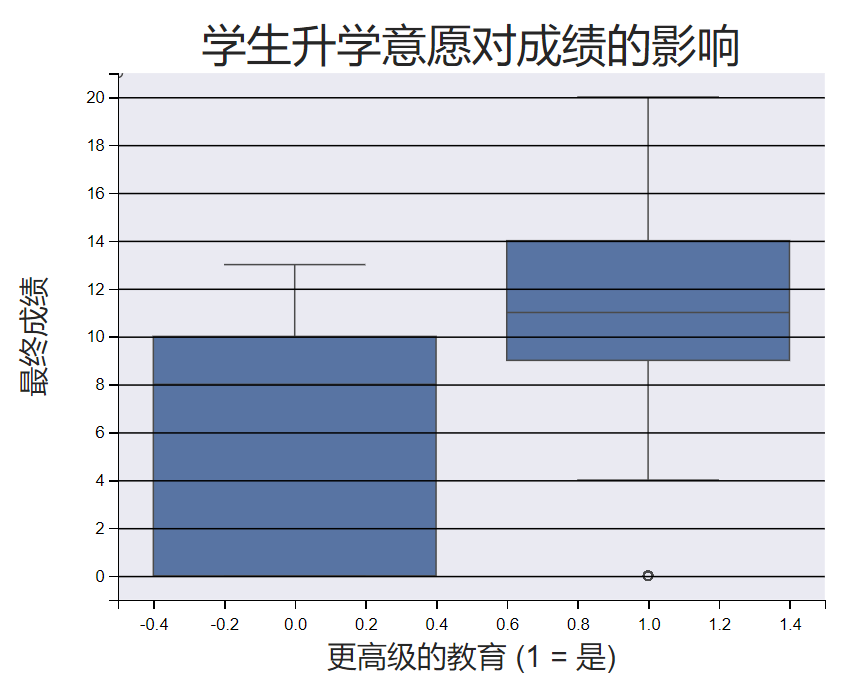
5.3.13 双亲受教育水平的影响





5.3.14 学生自己的升学意志对成绩的影响

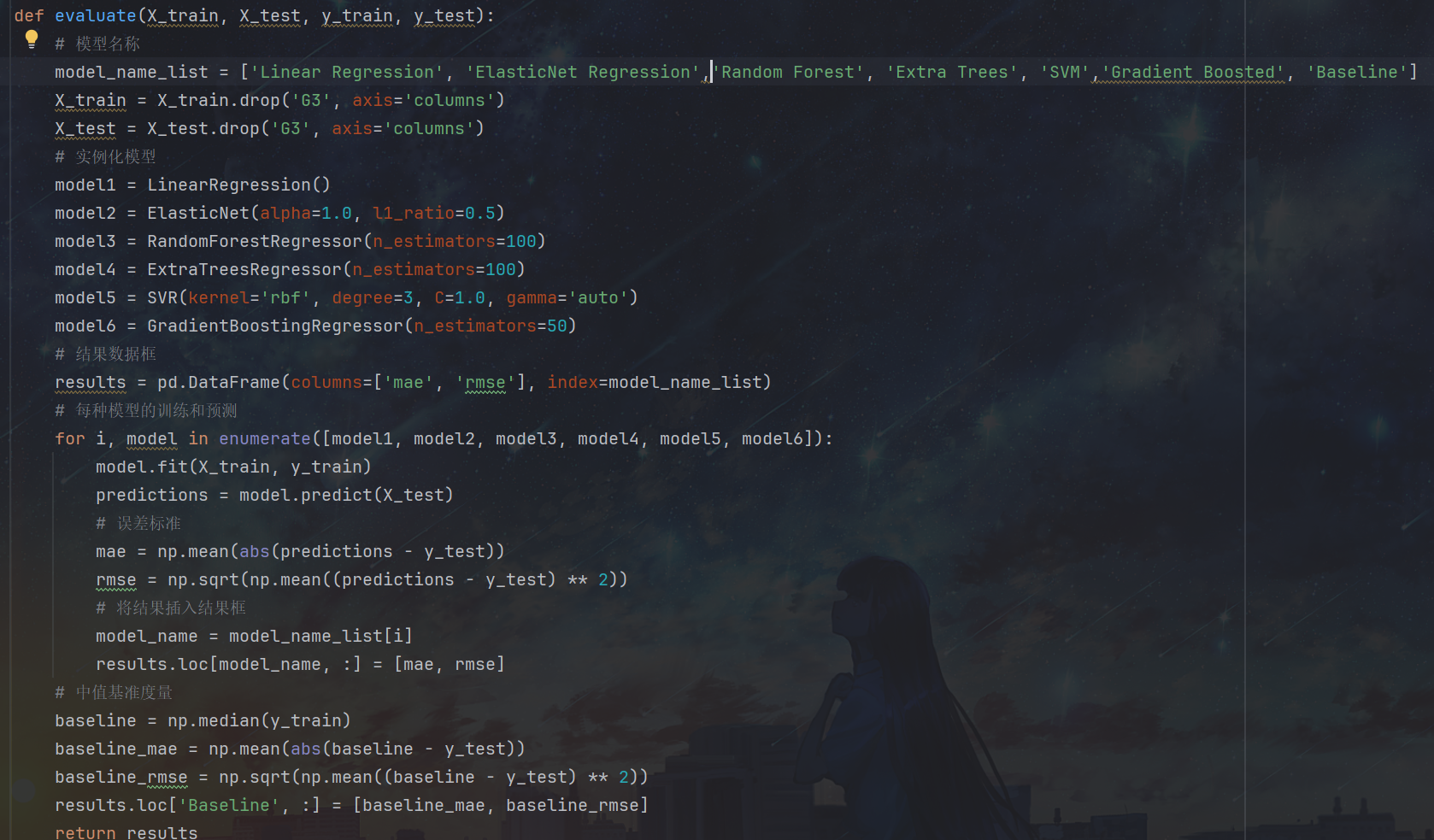


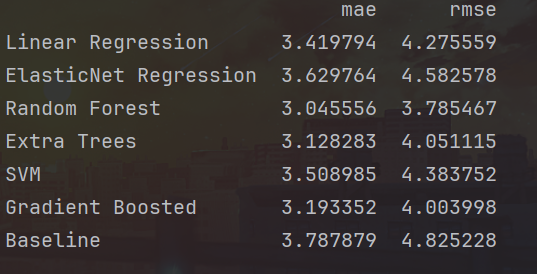


# 分割数据集  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(student, labels, test\_size=0.25, random\_state=42)  
  
  
# 计算平均绝对误差和均方根误差  
# MAE-平均绝对误差  
# RMSE-均方根误差  
def evaluate\_predictions(predictions, true):  
 mae = np.mean(abs(predictions - true))  
 rmse = np.sqrt(np.mean((predictions - true) \*\* 2))  
  
 return mae, rmse  
  
  
# 求中位数  
median\_pred = X\_train['G3'].median()  
  
# 所有中位数的列表  
median\_preds = [median\_pred for \_ in range(len(X\_test))]  
  
# 存储真实的G3值以传递给函数  
true = X\_test['G3']  
  
# 展示基准  
mb\_mae, mb\_rmse = evaluate\_predictions(median\_preds, true)  
print('Median Baseline MAE: {:.4f}'.format(mb\_mae))  
print('Median Baseline RMSE: {:.4f}'.format(mb\_rmse))

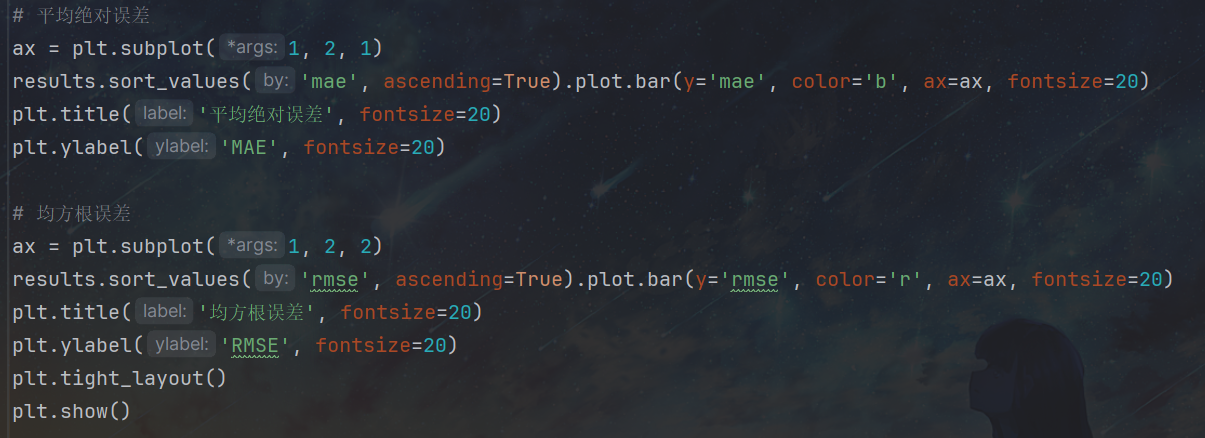


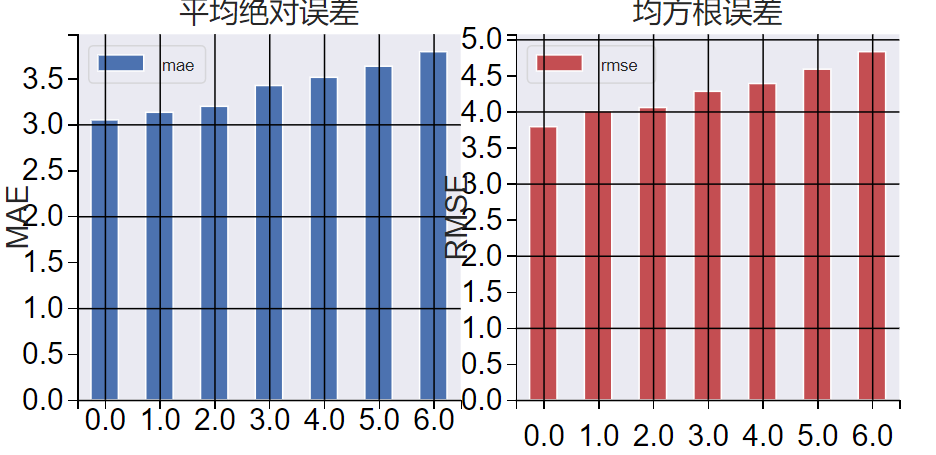
5.3.15 通过训练集训练和测试集测试来生成多个线性模型





5.3.16 得出平均绝对误差和均方根误差





**6.实验结论**

1. **学生成绩的多因素影响**：分析结果表明，学生的成绩受到年龄、城乡背景、失败次数、双亲受教育水平和学生升学意愿等多种因素的影响。这强调了教育过程中需要考虑的多维度因素。
2. **数据驱动的决策支持**：通过构建的线性回归模型，学校能够基于实证数据做出更加科学的决策，如调整教学方法、优化课程设置等。
3. **个性化学习路径的指导**：模型预测结果可以帮助学生根据自身情况，如家庭背景、学习习惯等，制定个性化的学习计划，以提高学习效率。
4. **教育公平性的促进**：研究揭示了不同社会经济背景下学生成绩的差异，为促进教育公平提供了依据，有助于学校和政策制定者采取措施，减少成绩差距。
5. **持续的模型优化**：虽然当前模型已经能够提供有价值的见解，但随着数据的积累和算法的改进，模型的准确性和预测能力有待进一步提升。
6. **教育实践的启示**：本研究为教育工作者提供了实证研究基础，有助于他们在教学实

践中更好地理解学生需求，实现教学内容和方法的创新。

预测学生成绩的目的是为了更好地理解影响学业表现的因素，并据此采取行动来提高学生的成绩。根据预测结果，为学生提供定制化的学习计划和辅导；对可能落后的学生进行早期识别和干预；根据学生需求调整教学策略和课程内容；为可能面临压力的学生提供心理健康支持等等方式。

**7 附录**

**7.1实验代码**

import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
from matplotlib.font\_manager import FontProperties  
from sklearn.linear\_model import LinearRegression  
from sklearn.linear\_model import ElasticNet  
from sklearn.ensemble import RandomForestRegressor  
from sklearn.ensemble import ExtraTreesRegressor  
from sklearn.ensemble import GradientBoostingRegressor  
from sklearn.svm import SVR  
from sklearn.model\_selection import train\_test\_split  
from sklearn.preprocessing import MinMaxScaler  
from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, median\_absolute\_error  
import scipy  
import pickle  
  
# 初始化数据  
plt.rcParams['font.sans-serif'] = ['SimHei'] # 中文字体设置-黑体  
plt.rcParams['axes.unicode\_minus'] = False # 解决保存图像是负号'-'显示为方块的问题  
sns.set(font='SimHei') # 解决Seaborn中文显示问题  
student = pd.read\_csv('student-mat.csv')  
#print(student.head())  
  
  
# 分析G3数据属性  
# print(student['G3'].describe())  
  
# 根据人数多少统计各分数段的学生人数  
grade\_counts = student['G3'].value\_counts().sort\_values().plot.barh(width=.9, color=sns.color\_palette('inferno', 40))  
grade\_counts.axes.set\_title('各分数值的学生分布', fontsize=30)  
grade\_counts.set\_xlabel('学生数量', fontsize=30)  
grade\_counts.set\_ylabel('最终成绩', fontsize=30)  
plt.show()  
  
# 从低到高展示成绩分布图  
grade\_distribution = sns.countplot(student['G3'])  
grade\_distribution.set\_title('成绩分布图', fontsize=30)  
grade\_distribution.set\_xlabel('期末成绩', fontsize=20)  
grade\_distribution.set\_ylabel('人数统计', fontsize=20)  
plt.show()  
  
# 检查各个列是否有null值，如果没有表示成绩中的0分确实是0分  
# print(student.isnull().any())  
  
# 分析性别比例  
male\_studs = len(student[student['sex'] == 'M'])  
female\_studs = len(student[student['sex'] == 'F'])  
print('男同学数量:', male\_studs)  
print('女同学数量:', female\_studs)  
  
# 分析年龄分布比例（曲线图）  
age\_distribution = sns.kdeplot(student['age'], fill=True)  
age\_distribution.axes.set\_title('学生年龄分布图', fontsize=30)  
age\_distribution.set\_xlabel('年龄', fontsize=20)  
age\_distribution.set\_ylabel('比例', fontsize=20)  
plt.show()  
  
# 分性别年龄分布图（柱状图）  
age\_distribution\_sex = sns.countplot(x='age', hue='sex', data=student)  
age\_distribution\_sex.axes.set\_title('不同年龄段的学生人数', fontsize=30)  
age\_distribution\_sex.set\_xlabel('年龄', fontsize=30)  
age\_distribution\_sex.set\_ylabel('人数', fontsize=30)  
plt.show()  
  
# 各年龄段的成绩箱型图  
age\_grade\_boxplot = sns.boxplot(x='age', y='G3', data=student)  
age\_grade\_boxplot.axes.set\_title('年龄与分数', fontsize=30)  
age\_grade\_boxplot.set\_xlabel('年龄', fontsize=20)  
age\_grade\_boxplot.set\_ylabel('分数', fontsize=20)  
plt.show()  
  
# 各年龄段的成绩分布图##########################################  
age\_grade\_plot = sns.stripplot(x='age', y='G3', data=student, size=4, jitter=True)  
age\_grade\_plot.axes.set\_title('年龄与分数', fontsize=30)  
age\_grade\_plot.set\_xlabel('年龄', fontsize=20)  
age\_grade\_plot.set\_ylabel('分数', fontsize=20)  
plt.show()  
  
# 城乡学生计数  
areas\_countplot = sns.countplot(student['address'])  
areas\_countplot.axes.set\_title('城乡学生', fontsize=30)  
areas\_countplot.set\_xlabel('家庭住址', fontsize=20)  
areas\_countplot.set\_ylabel('计数', fontsize=20)  
plt.show()  
  
# Grade distribution by address  
sns.kdeplot(student.loc[student['address'] == 'U', 'G3'], label='Urban', fill=True)  
sns.kdeplot(student.loc[student['address'] == 'R', 'G3'], label='Rural', fill=True)  
plt.title('城市学生获得了更好的成绩吗？', fontsize=20)  
plt.xlabel('分数', fontsize=20)  
plt.ylabel('占比', fontsize=20)  
plt.show()  
  
# 选取G3属性值  
labels = student['G3']  
# 删除school，G1和G2属性  
student = student.drop(['school', 'G1', 'G2'], axis='columns')  
# 对离散变量进行独热编码  
student = pd.get\_dummies(student)  
# 选取相关性最强的8个  
most\_correlated = student.corr().abs()['G3'].sort\_values(ascending=False)  
most\_correlated = most\_correlated[:9]  
print(most\_correlated)  
  
# 失败次数成绩分布图################################################  
failures\_plot = sns.stripplot(x='failures', y='G3', data=student, size=4, jitter=True)  
failures\_plot.axes.set\_title('失败次数少的学生分数更高吗？', fontsize=30)  
failures\_plot.set\_xlabel('失败次数', fontsize=20)  
failures\_plot.set\_ylabel('最终成绩', fontsize=20)  
plt.show()  
  
# 双亲受教育水平的影响  
family\_ed = student['Fedu'] + student['Medu']  
family\_ed\_boxplot = sns.boxplot(x=family\_ed, y=student['G3'])  
family\_ed\_boxplot.axes.set\_title('双亲受教育水平的影响', fontsize=30)  
family\_ed\_boxplot.set\_xlabel('家庭教育水平(Mother + Father)', fontsize=20)  
family\_ed\_boxplot.set\_ylabel('最终成绩', fontsize=20)  
plt.show()  
  
# 学生自己的升学意志对成绩的影响  
personal\_wish = sns.boxplot(x=student['higher\_yes'], y=student['G3'])  
personal\_wish.axes.set\_title('学生升学意愿对成绩的影响', fontsize=30)  
personal\_wish.set\_xlabel('更高级的教育 (1 = 是)', fontsize=20)  
personal\_wish.set\_ylabel('最终成绩', fontsize=20)  
plt.show()  
  
# 分割数据集  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(student, labels, test\_size=0.25, random\_state=42)  
  
  
# 计算平均绝对误差和均方根误差  
# MAE-平均绝对误差  
# RMSE-均方根误差  
def evaluate\_predictions(predictions, true):  
 mae = np.mean(abs(predictions - true))  
 rmse = np.sqrt(np.mean((predictions - true) \*\* 2))  
  
 return mae, rmse  
  
  
# 求中位数  
median\_pred = X\_train['G3'].median()  
  
# 所有中位数的列表  
median\_preds = [median\_pred for \_ in range(len(X\_test))]  
  
# 存储真实的G3值以传递给函数  
true = X\_test['G3']  
  
# 展示基准  
mb\_mae, mb\_rmse = evaluate\_predictions(median\_preds, true)  
print('Median Baseline MAE: {:.4f}'.format(mb\_mae))  
print('Median Baseline RMSE: {:.4f}'.format(mb\_rmse))  
  
  
# 通过训练集训练和测试集测试来生成多个线性模型  
def evaluate(X\_train, X\_test, y\_train, y\_test):  
 # 模型名称  
 model\_name\_list = ['Linear Regression', 'ElasticNet Regression','Random Forest', 'Extra Trees', 'SVM','Gradient Boosted', 'Baseline']  
 X\_train = X\_train.drop('G3', axis='columns')  
 X\_test = X\_test.drop('G3', axis='columns')  
 # 实例化模型  
 model1 = LinearRegression()  
 model2 = ElasticNet(alpha=1.0, l1\_ratio=0.5)  
 model3 = RandomForestRegressor(n\_estimators=100)  
 model4 = ExtraTreesRegressor(n\_estimators=100)  
 model5 = SVR(kernel='rbf', degree=3, C=1.0, gamma='auto')  
 model6 = GradientBoostingRegressor(n\_estimators=50)  
 # 结果数据框  
 results = pd.DataFrame(columns=['mae', 'rmse'], index=model\_name\_list)  
 # 每种模型的训练和预测  
 for i, model in enumerate([model1, model2, model3, model4, model5, model6]):  
 model.fit(X\_train, y\_train)  
 predictions = model.predict(X\_test)  
 # 误差标准  
 mae = np.mean(abs(predictions - y\_test))  
 rmse = np.sqrt(np.mean((predictions - y\_test) \*\* 2))  
 # 将结果插入结果框  
 model\_name = model\_name\_list[i]  
 results.loc[model\_name, :] = [mae, rmse]  
 # 中值基准度量  
 baseline = np.median(y\_train)  
 baseline\_mae = np.mean(abs(baseline - y\_test))  
 baseline\_rmse = np.sqrt(np.mean((baseline - y\_test) \*\* 2))  
 results.loc['Baseline', :] = [baseline\_mae, baseline\_rmse]  
 return results  
results = evaluate(X\_train, X\_test, y\_train, y\_test)  
print(results)  
  
# 找出最合适的模型  
plt.figure(figsize=(12, 8))  
  
# 平均绝对误差  
ax = plt.subplot(1, 2, 1)  
results.sort\_values('mae', ascending=True).plot.bar(y='mae', color='b', ax=ax, fontsize=20)  
plt.title('平均绝对误差', fontsize=20)  
plt.ylabel('MAE', fontsize=20)  
  
# 均方根误差  
ax = plt.subplot(1, 2, 2)  
results.sort\_values('rmse', ascending=True).plot.bar(y='rmse', color='r', ax=ax, fontsize=20)  
plt.title('均方根误差', fontsize=20)  
plt.ylabel('RMSE', fontsize=20)  
plt.tight\_layout()  
plt.show()  
  
# 保存线性回归模型  
model = LinearRegression()  
model.fit(X\_train, y\_train)  
filename = 'LR\_Model'  
pickle.dump(model, open(filename, 'wb'))

**7.2数据集的属性介绍**

1.学校-学生学校（二进制：“ GP”-加布里埃尔·佩雷拉（Gabriel Pereira）或“ MS”-Mousinho da Silveira）

2.性别-学生的性别（二进制：“ F”-女性或“ M”-男性）

3.年龄-学生的年龄（数字：15至22）

4.地址-学生的家庭住址类型（二进制：“ U”-城市或“ R”-农村）

5.famsize-家庭大小（二进制：“ LE3”-小于或等于3或“ GT3”-大于3）

6.Pstatus-父母的同居状态（二进制：“ T”-同居或“ A”-分开）

7.Medu-母亲的教育（数字：0-无，1-初等教育（四年级），2 – 5至9年级，3 –中等教育或4 –高等教育）

8.Fedu-父亲的教育（数字：0-无，1-初等教育（四年级），2 – 5至9年级，3 –中等教育或4 –高等教育）

9.Mjob-母亲的工作（名义：“教师”，“与健康”有关的，民事“服务”（例如行政或警察），“在家”或“其他”）

10.Fjob-父亲的工作（名义：“教师”，“与健康”相关的，民事“服务”（例如行政或警察），“在家”或“其他”）

11.理由-选择这所学校的理由（名义：接近“家”，学校“声誉”，“课程”偏好或“其他”）

12.监护人-学生的监护人（名词：“母亲”，“父亲”或“其他”）

13.traveltime-学校到学校的旅行时间（数字：1-<15分钟，2-15至30分钟，3-30分钟至1小时或4-> 1小时）

14.学习时间-每周学习时间（数字：1-<2小时，2-2至5小时，3-5至10小时或4-> 10小时）

15.失败-过去类失败的次数（数字：如果1 <= n ，则为n，否则为4）

16.schoolup-额外的教育支持（二进制：是或否）

17.famsup-家庭教育支持（二进制：是或否）

18.付费-课程主题内的额外付费课程（数学或葡萄牙语）（二进制：是或否）

19.活动-课外活动（二进制：是或否）

20.托儿所-上托儿所（二进制：是或否）

21.更高-想要接受高等教育（二进制：是或否）

22.互联网-在家上网（二进制：是或否）

23.浪漫-具有浪漫关系（二进制：是或否）

24.家族-家庭关系的质量（数字：从1-非常差到5-极好）

25.空闲时间-放学后的空闲时间（数字：从1-非常低到5-非常高）

26.外出-与朋友外出（数字：从1-非常低到5-非常高）

27.Dalc-工作日酒精消耗（数字：从1-非常低到5-非常高）

28.Walc-周末酒精消耗（数字：从1-非常低至5-非常高）

29.健康-当前的健康状况（数字：从1-非常差到5-非常好）

30.缺勤-缺勤人数（数字：0到93）

这些成绩与课程主题（数学或葡萄牙语）相关：

31.G1-第一期成绩（数字：0至20）

32.G2-第二学期成绩（数字：0至20）

33.G3-最终成绩（数字：0到20，输出目标）