本课题采用线性回归 (Linear Regression) 的方法得出学习动机、学业成绩是如何影响学生职业规划的。应用得出的模型，预测现阶段学生对未来职业的规划情况，并提早矫正职业规划不明确的学生，已到达让更多的学生有明确的职业规划的目标。

该项目利用问卷调查的方法收集了大学生学习动机，职业成熟度，学习成绩等多方面的数据作为样本。采用特征选取筛选出和因变量相关的变量作为线性回归的候选变量。线性回归的目的是分析自变量如何影响目标变量，并给出合理的拟合关系式，其整个主要流程如图1所示：



图1. 数据处理流程

*x*表示不同的自变量，比如学生的学习成绩和学习动机，*y*表示目标变量职业规划。通过在大量的训练集上进行学习，得到因变量和目标变量（职业规划）的拟合关系。此课题分为六步，详细细节将在以下分别描述：

**1. 目标变量选取**

职业规划：大学生职业问卷调查表分数（29个题，每题5分满分，分越高职业认知越高），用y表示。

**2. 自变量选取**

此课题有两个自变量，学习成绩和学习动机。学习成绩采用该受访者到目前为止的加权平均成绩表示，是0到100之间的一个整数，如89.5，用表示；根据研究方法的调查问卷结果，学习动机变量可以用1-20的量化数字表示，分值越大表明动机越弱，用表示。

**3. 目标变量和自变量之间的相关性**

选取两个自变量之后，我们需要进行特征选取，也就是分析它们和目标变量之间是否存在相关性。首先，我们分别画出这两个自变量和目标变量之间的分布图来直观的理解。通过图2，我们可以清楚地看出学习成绩和学习动机分别和职业规划成正相关和负相关。

为了更加精确的计算两个自变量和目标变量之间的相关性，我们采用皮尔森相关系数Pearson's correlation coefficient来计算。其计算公式如下：

其中，表示训练集样本大小，表示自变量 ， 表示样本中自变量因变量的平均值。通过样本计算，我们得出r(,y) = 0.08824，r(, ) = -0.2025. 此计算表明选取的两个自变量均和目标变量存在相关性，属于有效变量。

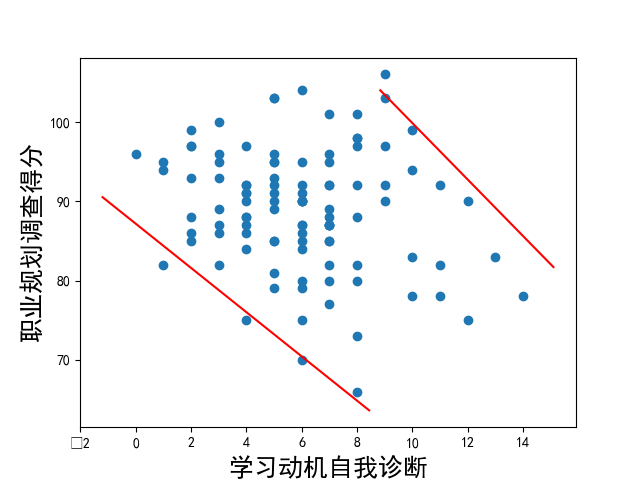
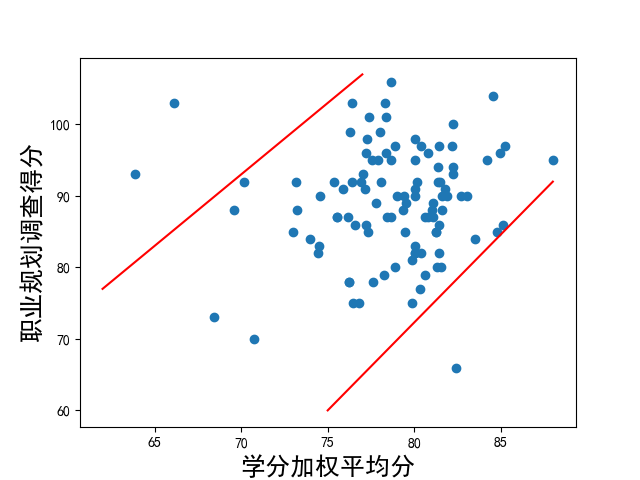


图2. 学习成绩，学习动机与职业规划的分布图

**4. 线性回归**

通过我们分别计算两个因变量和目标变量之间的皮尔逊相关系数，我们可以发现，两个变量和学生的职业规划都是相关的。为了进一步确定这两个变量是如何影响职业规划的，我们假设其服从线性关系，即. 我们采用双变量的线性回归，得出a b c三个参数的拟合值。具体计算细节，课参考附录中的代码。

通过计算我们得到：*a= 0.045，b=-0.621，c=88.923.*

**5. 模型准确性验证**

为了检测我们得出的 的有效性，我们把这个数据集分为训练集合和测试集。在此项目里，我们随机选取80%的数据作为训练集，在此进行线性回归分析，训练模型。在剩余的20%上进行测试，这里我们引入误差error来衡量模型的准确性。计算出模型预测的职业规划值和真实职业规划之间的误差来描述模型的准确性。进而分析我们提出的方案的有效性。

其中表示测试集中的样本个数，表示样本的真实结果，表示通过我们的模型的预测结果。计算结果显示表明我们的模型的预测结果和真实值的误差只有0.09. 可以很好的预测学生职业规划。

**6. 预测案例**

通过模型的准确性分析，表明我们的模型在预测学生职业规划的有效性。在未来，假设我们已知学生的学习成绩为75.89，学生的学习动机为4， 那么我们可以预测出其职业规划的数值为：0.045 \* 75.89 + 0.621 \* 4 + 88.923 = 94.82205. 此学生具有良好的职业规划，不需要进行校正。

附录：（代码）

**from** sklearn **import** preprocessing, cross\_validation, svm, metrics  
**from** sklearn.linear\_model **import** LinearRegression  
**from** numpy **import** \*  
**import** pandas **as** pd  
**import** matplotlib.pyplot **as** plt  
**from** pylab **import** \*  
mpl.rcParams[**'font.sans-serif'**] = [**'SimHei'**]  
data = pd.read\_csv(**'learn\_motivation.csv'**)  
print(corrcoef([data.score,data.self\_test,data.motivation]))  
  
**def** draw\_cof(data):  
 plt.scatter(data.score, data.motivation)  
 plt.plot([75, 88], [60, 92], c=**'r'**)  
 plt.plot([62, 77], [77, 107], c=**'r'**)  
 plt.xlabel(**'学分加权平均分'**, fontsize=18)  
 plt.ylabel(**'职业规划调查得分'**, fontsize=18)  
 plt.show()  
 plt.scatter(data.self\_test, data.motivation, )  
 plt.xlabel(**'学习动机自我诊断'**, fontsize=18)  
 plt.ylabel(**'职业规划调查得分'**, fontsize=18)  
 plt.plot([-1.2, 8.44], [90.52, 63.62], c=**'r'**)  
 plt.plot([8.84, 15.1], [104, 81.7], c=**'r'**)  
 plt.show()  
X = np.array(data[[**'score'**,**'self\_test'**]])  
y = np.array(data[**'motivation'**])  
  
X\_train, X\_test, y\_train, y\_test = cross\_validation.train\_test\_split(X, y, test\_size=0.2)  
clf = LinearRegression()  
clf.fit(X\_train, y\_train)  
a, b = clf.coef\_  
c = clf.intercept\_  
print(a,b,c)  
y\_pre = clf.predict(X\_test)  
mean\_sprt\_error = np.sqrt(metrics.mean\_squared\_error(y\_test, y\_pre))  
mean\_true = y\_test.mean()  
print(mean\_sprt\_error/mean\_true)