|  |  |
| --- | --- |
| 项目名称 | 学生成绩预测 |
| 项目代号 | Predict |
| 起始时间 | 2018-07-2 |
| 产品名称 | 学生成绩预测 |
| 产品代号 | Predict |
| 产品版本 | 1.7 |

学生成绩预测

|  |  |  |
| --- | --- | --- |
| **文档编号** | **文档类型** | **密级** |
| **P**redict\_1.7 |  |  |
| **编制/日期** | **审核/日期** | **批准/日期** |
| 邱光磊/2018-07-3 |  |  |
| **变更控制状态** | **对应实施阶段** | **页数** |
|  | 需求开发 | 共15 页 |

王佳腾工作小组

文档修改记录（Revision Chart）

\*S – START A - ADDED M - MODIFIED D - DELETED

| 版本号 | 日期 | 修订人 | S\* A M D | 版本修订描述 | 变更请求号 |
| --- | --- | --- | --- | --- | --- |
| 草稿v0.1 | 2018-7-3 | 邱光磊 | S | 首次编写。 | NA |
| 修订稿v1.0 | 2018-7-5 | 增嘉宇 | M | 定稿修订。 |  |
| 修订稿v1.5 | 2018-7-6 | 王佳腾 | M | 细节更正。 |  |
| 修订稿v1.7 | 2018-7-16 | 邱光磊 | M | 版本更正。 |  |

文档修订号：1，文档编辑总时间：0分钟。

# 目录

[学生成绩预测 1](#_Toc24684)

[王佳腾工作小组 1](#_Toc15779)

[文档修改记录（Revision Chart） 2](#_Toc4150)

[目录 3](#_Toc26825)

[1. 引言 4](#_Toc350)

[1.1 编写目的 4](#_Toc22306)

[1.2 背景 4](#_Toc9213)

[2. 任务概述 5](#_Toc3306)

[2.1 目标 5](#_Toc30167)

[2.2 用户特点 5](#_Toc22884)

[2.3 运行环境 5](#_Toc30573)

[2.4 开发工具 5](#_Toc6192)

[3． 应用功能需求描述 5](#_Toc1660)

[3.1 应用功能概述 5](#_Toc17997)

[3.2 应用运行界面 5](#_Toc12657)

[3.3 成绩预测功能介绍 8](#_Toc20526)

[3.3.1 学生成绩预测----无独有偶方法论 8](#_Toc29609)

[3.3.2 实现预测功能的步骤 8](#_Toc27028)

# 引言

## 编写目的

随着科技的发展，现在数据的功能已不仅仅是简单的记录信息。经过与多位学生成绩使用者进行全面深入地探讨和分析，并完成《学生成绩预测》市场的前期调查后，提出了这份软件需求分析说明书。

此需求分析说明书对《学生成绩预测》软件做了全面细致的用户需求分析，明确所要开发的应用软件应具有的功能，使系统分析人员及软件开发人员能清楚地了解用户的需求，并在此基础上进一步提出概要设计说明书和完成后续设计与开发工作。

本说明书的预期读者为客户、业务或需求分析人员、测试人员、用户文档编写者、项目管理人员。

## 背景

学生成绩是各大高校用于对学生学业评价的一项重要指标，截止到2016年，从教务系统中可直接获取的学生成绩信息最早可追溯到2004级，共21643名学生；其中具备有效成绩的学生基本范围：2013级~2016级，共4985名学生，因此有效成绩数目总数：243877条。

数据量虽远不及大数据的基本要求，但仍然拥有一定量的可挖掘的信息，根据以往成绩预测未来的成绩便是其中之一。

然而，目前没有一款软件能够对学生的成绩建立一个良好的模型，并从中通过数学方法进行对未来成绩的预测。因此特开发本应用软件，以弥补市面上成绩预测市场的缺失，提升关心此部分功能需求用户的满意度。

# 2. 任务概述

## 目标

建立一个能过根据用户以往成绩得出一个限定范围内的预测成绩的应用。

## 用户特点

对自己未来表现充满好奇，渴望根据可预测未来进行当下学习方式调整的大学生。

## 运行环境

本应用最终将以网页形式呈现给用户，任何能够浏览网页的设备均可使用。

## 开发工具

开发语言：Python 3.6 ，开发工具：Visual Studio Code。

# 应用功能需求描述

## 3.1 应用功能概述

打开浏览器后，输入http://thoreausjuice.imwork.net/predict/网址会出现一个文本框供输入学生学号，学生可在下拉菜单中分别选择，考试类型、学分，点击预测按钮，便可得出预测成绩。

## 应用运行界面

功能描述：

1. 打开浏览器后，输入http://thoreausjuice.imwork.net/predict/网址会出现一个信息输入界面，如图1所示。
2. 通过你的信息填写，选项选择，如图2所示。
3. 点击预测，得出待预测成绩，如图3所示。



图1 程序启动界面



图2 信息填写



图3 得出预测成绩

## 成绩预测功能介绍

### 学生成绩预测----无独有偶方法论

戈特弗里德·威廉·凡·莱布尼茨，这位举世闻名的德国数学家，哲学家曾说：“世界上不存在两片完全相同的树叶。”而在我的哲学观中，相同可能不存在，但相似或许不少。所以我认为人类并不孤独。这也是我成绩预测算法的核心思想：只要数据量足够大，我总能找到两个极为相似的人。

在整个预测算法中，一共存在三个重点项：

1. 用某种数据形式表示一个学生的成绩特征
2. 用某种方法寻找与之相似的另一个学生
3. 用某种算法从匹配的信息中预测下一次考试的成绩

### 实现预测功能的步骤

**第一步----构建成绩矩阵：**

首先解决第一大项，我认为能够影响一个学生成绩的所有因素在该学生每次的考试中都将进行作用，成绩是每次综合影响的结果，每个因素每次作用的大小可能不同，但决定性因素变化不大。因此最开始，我尝试将每个学生的每次成绩放在一个巨大的矩阵中，将这个矩阵当作图像进行去噪，平滑，突出每次的决定性影响因素。但问题随之而来。

经过我对学生成绩的初步人工分析，我们手中的数据存在以下特点：

1. 由于数据来源问题，数据仅有当前在校生的成绩，这意味着不同年级不同专业的学生成绩数量都不是一个常数，因此逐一放入矩阵会造成部分成绩无法进行有效匹配。
2. 所有的成绩具备三种标识：学科类型和考试类型及学分

基于以上特点，我对学生成绩矩阵进行了重新构建，以考试类型及相应的学科类型分行，以该类型课程的学分数分列，矩阵构建如下：



这个矩阵代表了每个学生的成绩特征，相应的所有学科成绩通过某种算法填充进来。这个矩阵的好处是，不论一个学生有多少科成绩，都逃不出该矩阵的束缚，且所有成绩都能在匹配中发挥有效的作用。

接下来一个很关键的问题就是用什么数学方法能比较好的用相互关联的成绩得出一个新的标识成绩用以填入这只能容纳一个数的格子里。

没错，和你想的一样，我最开始也是尝试使用平均成绩，但效果非常不理想。

原因有以下几点：

1. 由于不同科目采取的不同计分法，有的学生会存有“不及格”字样的成绩，该种成绩无法进行有效的百分制转化，导致我们仅能使用0分进行替代，然而0分成绩无法体现一个学生在该次考试中的影响因素信息，与缺考无异，导致该数据失去分析价值。非但不会给出有效的匹配因素，甚至由于成绩过低导致平均成绩失真。在这一点上，经过验证，过高的成绩和过低的成绩在取平均值时，都会相当大的决定因素。这在神经网络的池化中已有结论。
2. 排除0分成绩的影响后，在后续的矩阵匹配中，会出现一种很有意思的现象，求得的两个相似矩阵在对相应学科的成绩预测中会出现待预测学生有补考成绩，匹配学生无补考成绩，即一方成绩为0的情况，该情况下，进行平均取值会导致预测值的必然错误。

在综合考虑了所有非常态情况后，我创造了一种全新的评估方法，我将之称为极化均值法。

首先原有的平均值算法是用成绩乘以数量的和除以数量得出平均值：

以两科成绩举例：

在极化均值中，我给每个成绩加了一个权值，权值的大小取决于成绩的高低，我认为分数越高其中能体现出来的有用信息越多，因而，在传统的平均值法中，我们可以认为这个权值为1，而成绩的数量即为权值的总和：

现在我们将这个默认的权值进行修改，将之与成绩关联起来，权值的大小为成绩的百分数，即现有成绩除以满分：

该方法最大的好处是在遇到0分的成绩时，均值成绩不再受无效值影响：

在此基础上，我们对该方法进行公式化并整理：

这便是加权计算后的均值，总体上来看是让大值留更多，小值保留更少，呈一种两极分化的趋势，因而将其命名为极化均值。

**第二步----进行学生匹配：**

从已有数据集中寻求与之匹配的学生，实际上是在做一个分类器，类似在一个约会网站上匹配两个人。在我为数不多的机器学习知识中，这方面效果比较好的方法非k-近邻莫属。

简单介绍一下k-近邻分类算法（kNN），k-近邻算法采用测量不同特征值之间的距离进行分类。它的工作原理是：存在一个样本数据集合，也称作训练样本集，并且样本集中每一个数据都存在标签，即我们知道样本集中每一个数据与所属分类的对应关系。输入没有标签的新数据后，将新数据的每个特征与样本集中数据对应的特征进行比较，然后算法提取样本集中特征最相似数据（最近邻）的分类标签。一般来说，我们只选择样本数据集中前k个最相似的数据，这就是k-近邻算法中k的出处，通常k是不大于20的整数。最后选择k个最相似数据中出现次数最多的分类，作为新数据的分类。

这是传统的k-近邻方法，传统k-近邻采用欧式距离进行匹配，其特点如下：

优点：精度高、对异常值不敏感、无数据输入假定。

缺点：计算复杂度高、空间复杂度高。

适用范围：数值型和标称型。

计算两个点A(，)，B(，)之间的距离公式为：

在北京理工大学数据智能实验室跟优秀的学长们学习手写数字识别的时候，我发现了一种全新的距离计算法，我将之成为特征向量法，精度高于传统k-近邻，无需调参，取最接近的单一结果即可，该方法求取每个手写数字的图像矩阵的特征向量，将特征向量视为该数据的特征，求取夹角最小的两个向量，进而实现最近邻匹配。当年在实验室轰动一时，主要也是大家都不知道，后来回本校和林博士交流后才知道这实际上是一种降维技术，前人将该种降维方法称为主成分分析（Principal Component Analysis，PCA）。

对数据进行简化除了能降低很多算法的计算开销，重点是还能去除噪声，前面我们说到这个成绩矩阵是存在噪声的，去噪是我一直比较想做的事情。

PCA，主成分分析具有如下特点：

优点：降低数据的复杂性，识别最重要的多个特征。

缺点：不一定需要，且有可能损失有用的信息。

适用数据类型：数值型数据。

通过PCA进行降维处理，我们就可以同时获得SVM和决策树的优点：一方面，得到了和决策树一样简单的分类器，同时分类间隔和SVM一样好。另外，由于只需要考虑一维信息，因此数据就可以通过比SVM简单得多的很容易采用的规则进行区分。而另一维信息只是对分类缺乏贡献的噪声数据。在二维平面下，这一点看上去微不足道，但是如果在高维空间下则意义重大。

通过数据集的协方差矩阵及其特征值分析，我们就可以求得这些主成分的值。

特征值分析是线性代数中的一个领域，它能够通过数据的一般格式来揭示数据的“真实”结构，即我们常说的特征向量和特征值。在等式中，是特征向量，是特征值。特征值都是简单的标量值，因此代表的是：如果特征向量被某个矩阵A左乘，那么它就等于某个标量乘以。

一旦得到了协方差矩阵的特征向量，我们就可以保留最大的N个值。这些特征向量也给出了N个最重要特征的真实结构。

将数据转换成前N个主成分的伪码大致如下：

去除平均值

计算协方差矩阵

计算协方差矩阵的特征值和特征向量

将特征值从小到大排序

保留最上面的N个特征向量

将数据转换到上述N个特征向量构建的新空间中

有关PCA这一部分摘自Harrington,P.的《机器学习实战》，遗憾的是在实际应用中，我并未按照此方法，原因如下：

1. 矩阵（方阵）在计算特征向量时要求矩阵呈正方形，我设计的为5×6的成绩矩阵并不完全符合要求。
2. 最后一个决定性的因素是时间，在之前对所需算法的调研与建立时耗费了大量的时间，致使最终没能完善PCA的具体实现。

尽管如此，我依旧不愿放弃降维的思路，所以对数据进行了简单的人工降维，即将5×6的矩阵逐行展开成1×30的一维向量，并直接计算向量间的最小角度得出最近邻向量，从而判断出与待预测学生成绩矩阵最为相似的另一个学生的成绩矩阵。

暂且认为这是一个低配版PCA，我将1×30的向量看作为原矩阵的特征向量，计算与之夹角最小的向量，使用向量间夹角的公式：

在使用该方法时，由于整体数据量的不足，仅取最小角度值所匹配的成绩矩阵。读者如果有能力有兴趣且具备足够的数据样本，推荐严格使用PCA原文所述过程，最终预测正确率应该会有不小的提升。

**第三步----预测下次成绩：**

毫无疑问这是一个难题，在所有步骤中这是最让我头疼的问题。我考虑过平均值，中位数，众数等等，最初完成算法的时候，还没能提出极化均值法，使用的就是传统的平均值进行逆运算，导致正确率奇低。这也直接催生了加权平均值的想法，最终得出了极化均值法，并改写了我整个程序的求值思路。

因而在后来的修正中，基本预测法变得简单，将两个矩阵待预测位置的成绩求取极化均值作为原始预测成绩，之后进行成绩修正：

1. 学生能力判断：

如果待预测学生的所有成绩中均没有补考、重考及重修的课程，我们认为该学生为优秀学生，对其原始预测成绩加5分。

如果待预测学生的所有成绩中补考、重考及重修的课程大于6门，我们认为该学生为非高分学生，对其原始预测成绩减5分。

1. 总成绩回归：

如果修正后的原始成绩超过满分，则将其降至满分减预测半区间。

如果修正后的原始成绩低于0分，则将其升至0分加预测半区间。

最后是预测值的区间设置，我评测预测值正确与否的标准是设定一个半区间长度，半区间长度为5时，预测区间为预测值的正负5 。

这便是笔者暂时所能想到的主要因素，以上便是整个预测算法最终的预测实现。在后期的调试中，我尝试加入一个-1到1的随机值，让预测值实现一个小规模的浮动，从而使预测的正确率浮动，的确会造成一定的几率的正确率提升，但正确率下降的风险一样存在，并未经过具体的统计，但预测次数足够多的情况下，上下浮动的此数应该相等。

有一个备选因素未能在最终程序中实现，其想法如下：

根据该学生优秀的成绩个数及不及格的成绩个数适当移动区间

优秀成绩占比高则将区间右移，反之则左移

移动量为优秀成绩数量或不及格成绩数量占此两种成绩个数和之比

举个例子：

以预测分数80为例，检测到该学生95分（优秀）成绩个数为15，不及格成绩个数为5，预测区间总长为10，则原有预测区间为75~85，修正后区间移动比为15 ÷ (15 + 5) = 0.75，即区间右移，偏移量为0.75 × 10 = 7.5 ，最终修改后的预测区间为77.5~87.5。

由于该想法未能在程序中实现，所以对预测结果的影响未知。

在整个预测算法中，使用的数据集情况如下：

能够提供的学生基本信息范围：2004级~2016级（共21643名学生）

具备有效成绩的学生基本范围：2013级~2016级（共4985名学生）

有效成绩数目总数：243877条

从中随机挑选出100名学生作为待预测学生

待预测学生成绩数目总数：6906

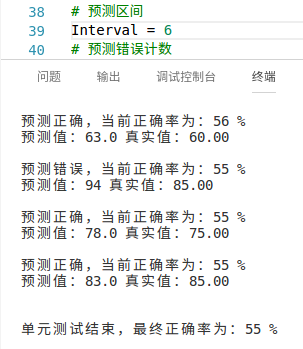
从待预测学生成绩中隐藏的成绩数：100

附两组不同区间下的预测正确率：（未加随机数浮动）

预测半区间：6（即预测值与真实值的差值不超过6）

正确率：55%

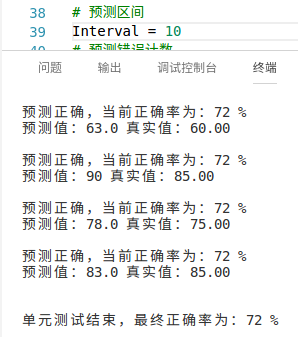
结果截图：



预测半区间：10（即预测值与真实值的差值不超过10）

正确率：72%

结果截图：



最后附上不同预测半区间与预测正确率的图表：

**附录：**

1. **经过数据清洗后的学生成绩样本截图：**



1. **参考文献（书籍）：**

机器学习实战 /（美）哈林顿（Harrington, P.）著；李锐等译.——北京：人民邮电出版社，2013.6（2017.3重印）