用机器学习预测高考成绩！？

ButtersPC

主业拳击手，副业分析师

只要数据集足够大，特征足够多，连你未来儿子是不是gay我都预测给你看！

什么？？机器学习已经可以用来预测高考成绩了？？？

理论上是绝对可行的！只要数据集足够大，特征足够多，连你未来儿子是不是gay我都预测给你看！可惜高考分数数据集从不公布，一般人也无法获取，但幸运的是我们有幸得到一份“某年上海市某区高考二模成绩”数据集，数据集质量很高，但也有缺失，我们将对其中缺失的考试成绩进行预测。

导读：

1. 本文展现了一个入门级别的数据分析项目，选用的工具为python，非常适合菜鸟们茶余饭后加餐品味
2. 老鸟们可以放肆批判代码，但希望能够对分析逻辑多加指点
3. 本文分为两部分：a. 对于数据进行描述性分析 b. 用机器学习预测缺失的成绩
4. 数据集选用“某年上海某区高考二模成绩”，对于人名均做脱敏处理，请放心食用
5. 上海过去几年的高考制度采用“3+1+1”的模式，语数英 + 加一学科 + 综合，加一学科指在物理、化学、生物、地理、政治、历史中选其一。***这条信息很重要，有助于帮我们理解数据**
6. 没有了
7. 真的没有番号

一、导入数据

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

path = r'***'    #此处为脱敏处理，实则是数据文件路径

#df_w是文科生原始数据
#df_l是理科生原始数据
df_w = pd.read_excel(path,sheet_name=0)
df_l = pd.read_excel(path,sheet_name=1)
```

In [2]: df_w.head()

Out[2]:

	学科	学校	准考证号	姓名	语文	数学	英语	五门调整总分	总分排名
0	政治	吴淞中学	11020311098		102.0	110.0	127.5	459.1	8
1	政治	吴淞中学	11020311097		102.0	101.0	114.5	452.7	15
2	政治	吴淞中学	11020311096		95.0	108.0	128.5	448.3	19
3	政治	吴淞中学	11020311104		107.0	96.0	116.0	446.6	23
4	历史	吴淞中学	11020321126		103.0	97.0	121.5	445.9	26

In [3]: df_l.head()

Out[3]:

	学科	学校	准考证号	姓名	语文	数学	英语	加一调整分	五门总分	总分排名
0	物理	行知中学	11010141001		109.0	127.0	123.5	117	494.9	38
1	物理	行知中学	11010141002		98.5	103.5	128.5	111	461.3	189
2	物理	行知中学	11010141003		102.0	135.0	123.0	127	507.0	21
3	物理	行知中学	11010141004		91.0	145.0	124.5	142	521.9	7
4	物理	行知中学	11010141005		87.0	136.5	130.0	144	519.9	8

总体上来说数据质量很高，非常工整。当然也存在几个问题：

1. 文科生数据表“五门调整总分”& 理科生数据表“五门总分” 有所不同 (简单的修改即可)

2. 文科生数据表中没有“加一调整分”(尝试通过机器学习预测)

二、简单的预处理

```
#两个df列名有所不同，修改后合并
df_w = df_w.rename(columns = {'五门调整总分':'五门总分'})
df = pd.concat([df_w,df_l],ignore_index= True)

#添加一列“三门总分”
sanmen = df.数学 + df.语文 + df.英语
df.insert(0,'三门总分',sanmen)
```

```
df.head()
```

	三门总分	五门总分	准考证号	加一调整分	姓名	学校	学科	总分排名	数学	英语	语文
0	339.5	459.1	11020311098	NaN		吴淞中学	政治	8	110.0	127.5	102.0
1	317.5	452.7	11020311097	NaN		吴淞中学	政治	15	101.0	114.5	102.0
2	331.5	448.3	11020311096	NaN		吴淞中学	政治	19	108.0	128.5	95.0
3	319.0	446.6	11020311104	NaN		吴淞中学	政治	23	96.0	116.0	107.0
4	321.5	445.9	11020321126	NaN		吴淞中学	历史	26	97.0	121.5	103.0

预处理后的概览

简单预处理之后，我们把文科生成绩和理科生成绩的两张数据表合并了，由于文科生数据表中没有“加一调整分”一列，因此所有文科生的该列为空（NaN），然后我们可以进入进一步挖掘了。

三、基础的描述性统计

在这一章节我们将通过对数据集进行描述，从而尽可能地探求数据中包含的有价值信息：

首先我们对“学科”特征进行分组（groupby），来看一下各科目的分班情况：

```
#分组后通过对准考证的计数来查看各科目的考生人数
g_xueke = df.groupby('学科')
g_xueke['准考证号'].count().sort_values(ascending=False)
```

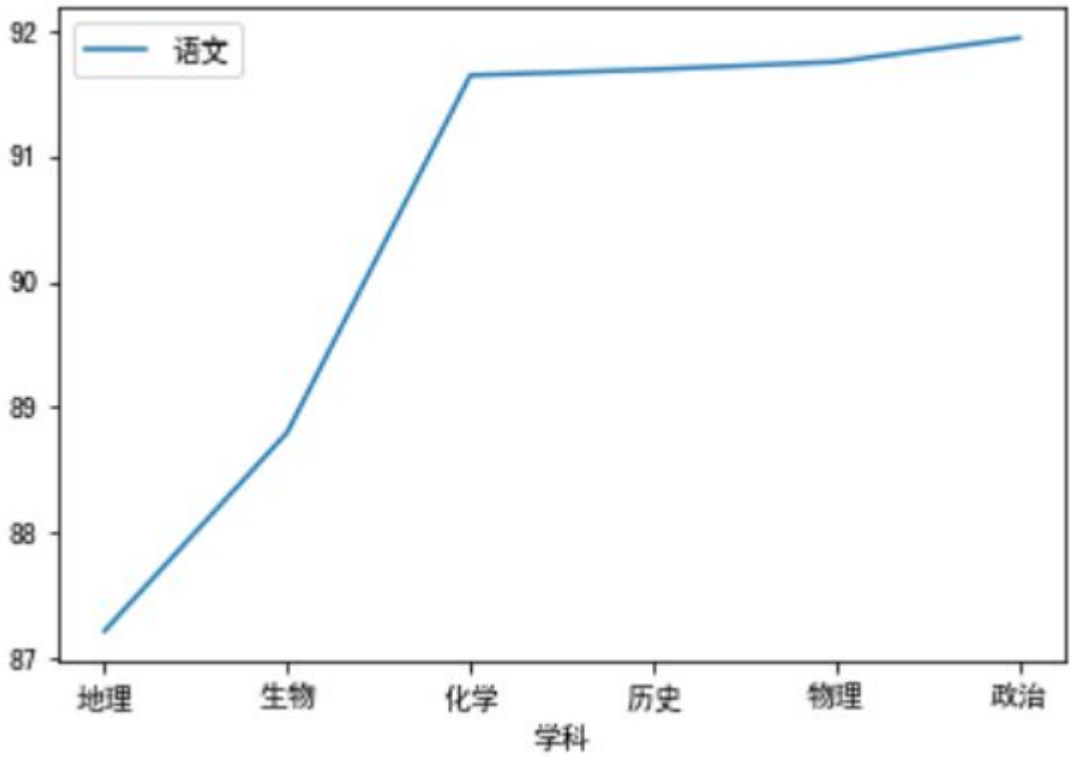
```
学科
化学    1072
物理     758
历史     383
生物     261
政治     212
地理     136
Name: 准考证号, dtype: int64
```

各科目考生人数，化学考生竟然最多？我曾经一直以为是物理

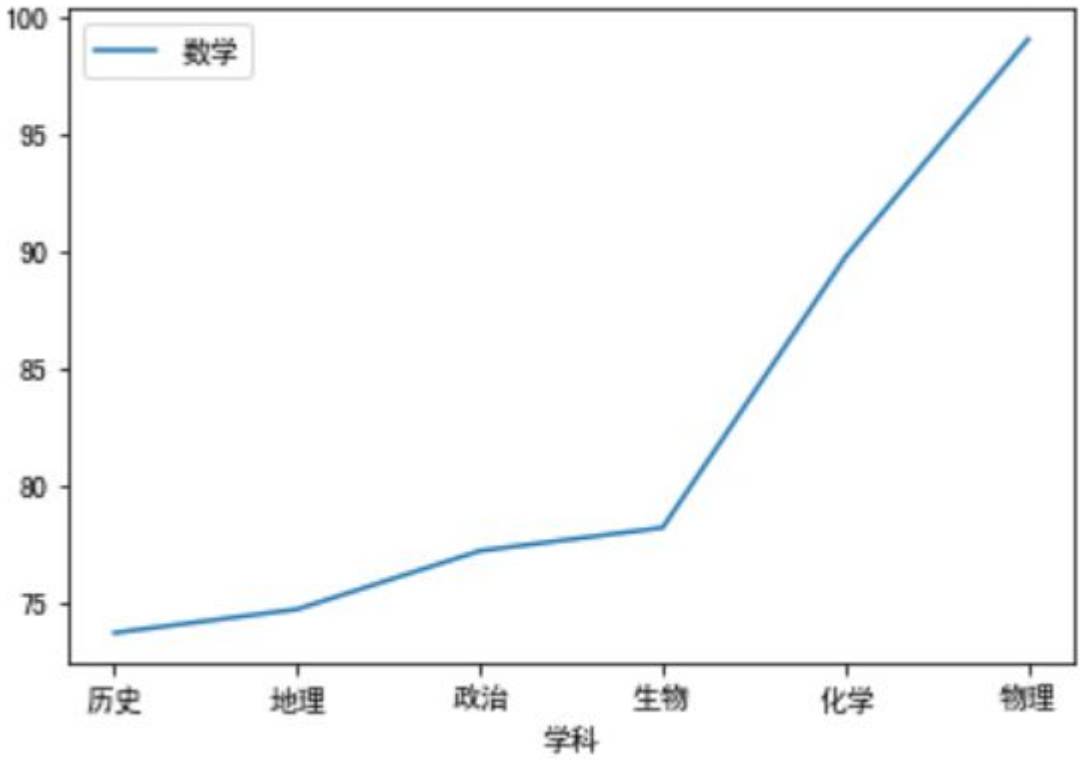
紧接着，我们用matplotlib，对各学科考生的平均分状况进行一个大致的判断：

```
g_xueke[['语文']].mean().sort_values(by='语文',ascending=True).plot()
g_xueke[['数学']].mean().sort_values(by='数学',ascending=True).plot()
g_xueke[['英语']].mean().sort_values(by='英语',ascending=True).plot()
```

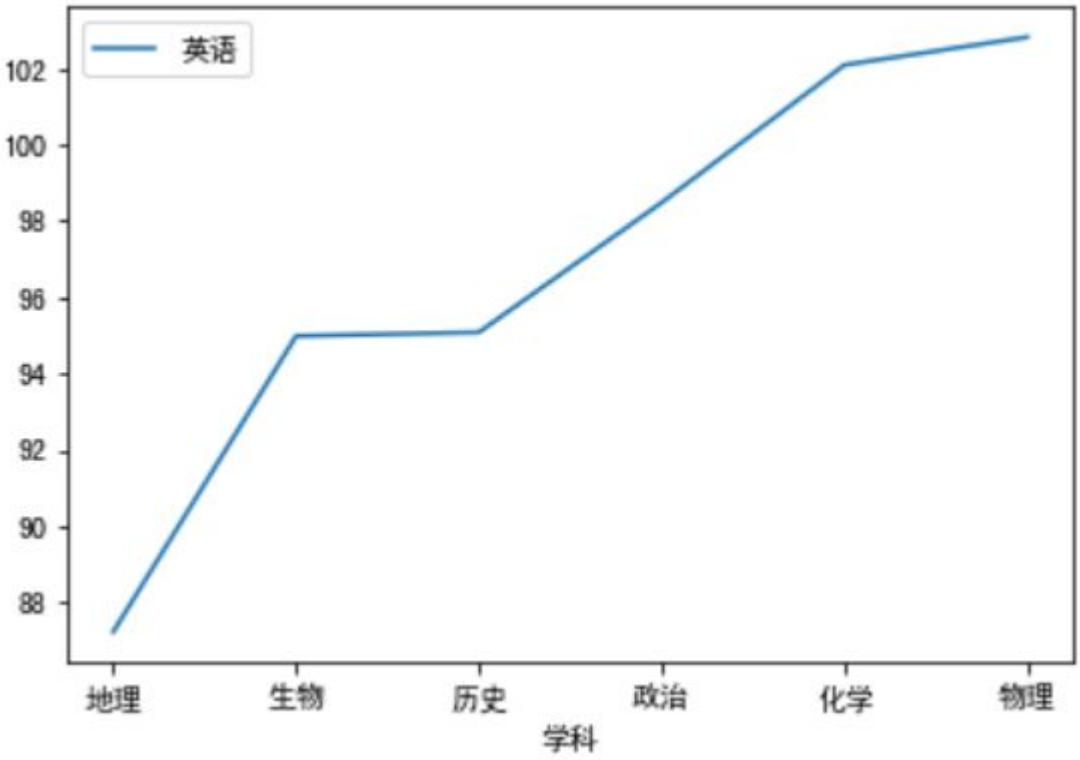
```
g_xueke[['三门总分']].mean().sort_values(by='三门总分',ascending=True).plot()
g_xueke[['五门总分']].mean().sort_values(by='五门总分',ascending=True).plot()
```



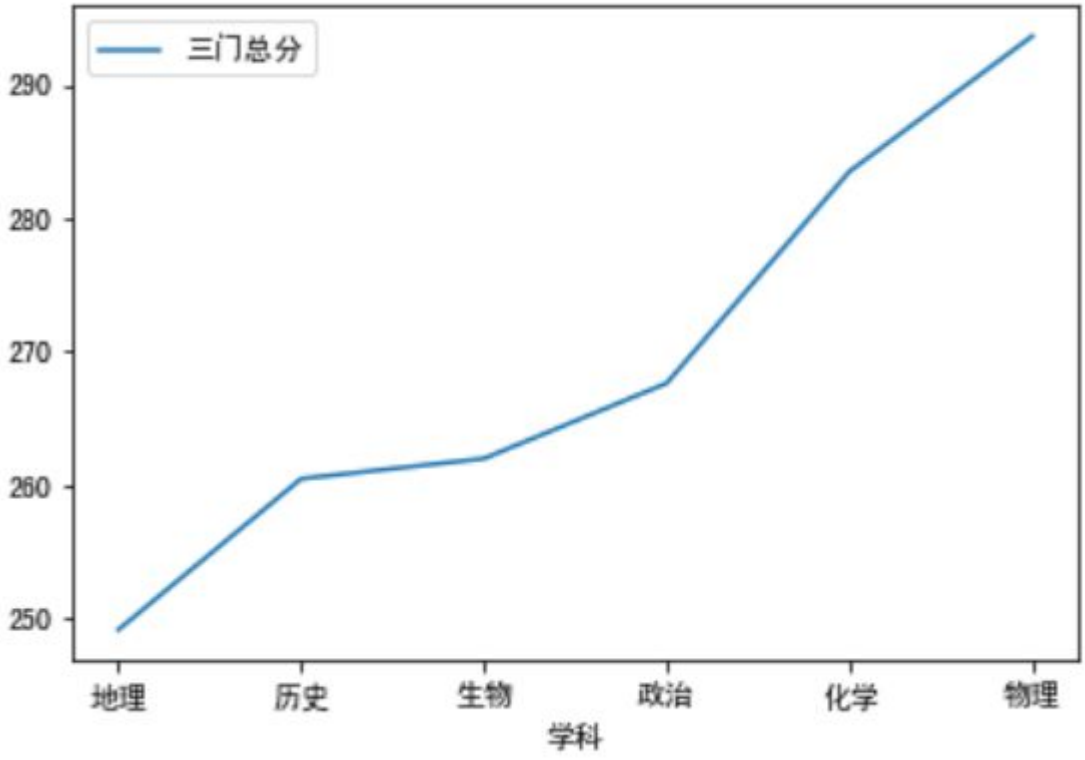
政治考生语文平均分最高（为什么不是历史[○･｀Д´･○]）

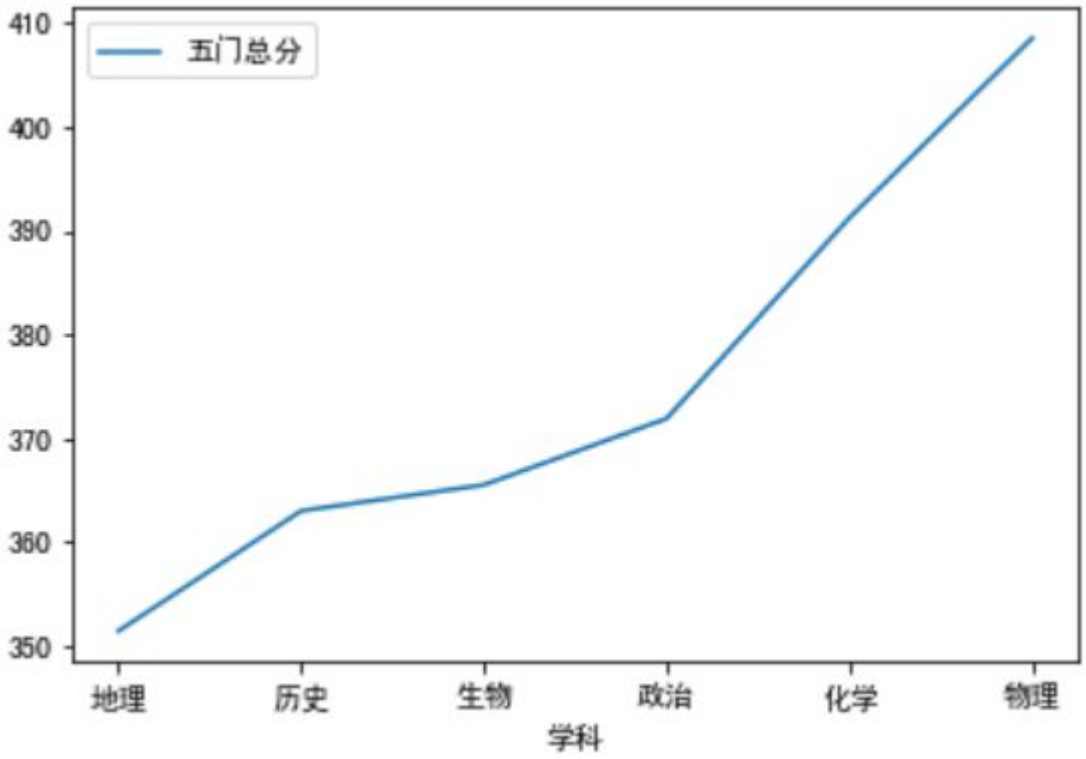


物理考生数学平均分最高，似乎很合理



物理考生英语平均分最高，这是为什么？





1. 选化学的学生最多，其次物理
2. 政治班语文平均分最高，物理班其次
3. 物理班数学平均分最高，化学班其次
4. 物理班英语平均分最高，化学班其次
5. 物理班五门总分平均成绩最高，化学班其次

总体来说物理班的总体实力最强，一般来说最顶尖的学生都会选物理（不知道这个背后有什么样的原因？），上海这边普遍是这样，不知道各个地区会不会有所不同？

再来，我们对“学校”特征进行分组（groupby）：

```
#按照学校分组，并查看各学校学生数量
g_xuexiao = df.groupby('学校')
g_xuexiao['准考证号'].count().sort_values(ascending=False)
```

学校
行知中学 520
上大附中 447
吴淞中学 352
罗店中学 310
宝山中学 287
顾村中学 168
通河中学 167
行知实验 136
高境一中 104
和衷高中 100
淞浦中学 99
上大附中新疆部 66
同洲模范 26
行中中学 22
建峰高中 18
Name: 准考证号, dtype: int64

这边我们发现了3组异常值：

同洲模范、行中中学、建峰高中三所学校的人数比较奇怪，只有20人左右

查阅网上资料后了解到：

1. 行中中学是行知中学放置借读生名额的学校（为了不影响升学率的手段）
2. 建峰高中是一所高职，其中会参加高考的学生并不多，因此参加高考模拟考的学生也不多
3. 许多人指责同洲模范高中部很差（似乎初中部还不错？），可能校方有意收缩高中部招生

同样地，我们也用matplotlib作图，对各学校的平均成绩做一个大致的判断（此处图就省略了）：

```
g_xuexiao[['语文']].mean().sort_values(by='语文',ascending=True).plot()
g_xuexiao[['数学']].mean().sort_values(by='数学',ascending=True).plot()
g_xuexiao[['英语']].mean().sort_values(by='英语',ascending=True).plot()
g_xuexiao[['三门总分']].mean().sort_values(by='三门总分',ascending=True).plot()
g_xuexiao[['五门总分']].mean().sort_values(by='五门总分',ascending=True).plot()
```

得到的结果是：

1. 语文平均分第一：罗店中学
2. 数学平均分第一：行知中学
3. 英语平均分第一：吴淞中学
4. 三门平均分第一：行知中学
5. 五门平均分第一：行知中学

有趣的是，行知中学数据平均分第一，并且也获得了三门平均分第一和五门平均分第一

自此我产生了一个疑问：数学考得好，更容易拿更高的名次？

尝试验证一下，通过皮尔森相关系数验证一下（皮尔森相关系数能够体现两组数据之间的线性关系）：

```
#分析各科成绩与总分之间的相关系数
df[['数学','语文','英语','三门总分','五门总分']].corr()
```

	数学	语文	英语	三门总分	五门总分	总分排名
数学	1.000000	0.370773	0.544709	0.867267	0.858378	-0.487797
语文	0.370773	1.000000	0.503907	0.653351	0.617937	-0.476315
英语	0.544709	0.503907	1.000000	0.856589	0.824188	-0.540952
三门总分	0.867267	0.653351	0.856589	1.000000	0.972618	-0.613766
五门总分	0.858378	0.617937	0.824188	0.972618	1.000000	-0.645787

红色框内是总分与各组数据的相关系数，一般来说>0.5就存在比较弱正相关性

果然数学与总分的相关系数最高，英语略低于数学，语文最低，而且低很多。

（窗帘为什么是蓝的？为什么眼中会有诡异的光？[○･｀Д´･ ○]，我一度觉得语文这科的评分标准就是为了让高考制度保留一定的自由裁量权，从而可以方便进行一些差别待遇）

我们进一步观测语数英成绩这三组数据：

```
#进一步探究这组数据
df[['数学','语文','英语']].describe()
```

	数学	语文	英语
count	2822.000000	2822.000000	2822.000000
mean	87.368179	91.223600	99.694011
std	21.731483	9.186677	18.098783
min	0.000000	0.000000	18.000000
25%	73.000000	86.000000	90.000000
50%	88.000000	92.000000	102.000000
75%	103.000000	97.000000	113.000000
max	147.000000	120.000000	141.000000

count为计数、mean为平均数、std为标准差

我们发现数学的标准差最大，这说明了数学成绩离散程度比较高，每一分的考生人数比较少，因此：数学每提高1分能干掉的其他考生比其他科目少，所以似乎看来提高数学成绩是收益最低的?

的确，数学没提高1分，对于考生在数学考试上的排名提升效果较小，但是在实践考试中我们是按照语数英三门科目加总后进行排名的（而且在加总的时候不分配权重），这就说明数学的1分，英语的1分，语文的1分，在最终三门总排名上的影响是等价的。

我们再回到数学标准差较大的讨论上，我认为数学成绩方差高是一种现象，导致这种现象的是因为数学的评分机制的关系：数学题不会就是不会，经常会出现一道大题20份只拿1分的情况（解：+1分），而语文不会做也能瞎诌诌，英语则题量比较大，得分点分配的很散。再进一步说，就是方差高说明考生在该项考试中成绩差距显著，考生们更容易在该项考试中被拉出等第，也更容易方便学校来选拔考生（之前复旦大学自主招生千分考项目，应该也是同样的思路，高方差为了更便于筛选考生）。

举个例子就是说：假设某个考生A，数学略高于平均水平，英语语文略低于平均水平，在数学考试高方差的放大下，最后考生A的三门总分排名可能是高于平均水平的。而考生B，语文略高于平均水平，数学英语略低于平局水平，在语文考试低方差的影响下，最后考生B的三门总分排名往往是低于平均水平的。

最终我得出的结论是，如果将考试分为n门，理论上来说这n门的最终成绩的方差应该是近似的，如果其中有一门的方差偏高，说明这一门更能凸显考生之间的差距（也就是可以拉开排名上的差距），如果作为考生，应该在方差大的科目上多努力，这样有助于最后提高排名（当然是在最终成绩不加权的情况下）。

（上述分析感谢@吴烜圣的点评，帮助我打开了思路）

除了方差，这组数据描述中还有一项值得关注的地方：那就是最小值

	数学	语文	英语
count	2822.000000	2822.000000	2822.000000
mean	87.368179	91.223600	99.694011
std	21.731483	9.186677	18.098783
min	0.000000	0.000000	18.000000
25%	73.000000	86.000000	90.000000
50%	88.000000	92.000000	102.000000
75%	103.000000	97.000000	113.000000
max	147.000000	120.000000	141.000000

count为计数 mean为平均数 std为标准差

其中有最小值0的数据，一般来讲考试不太会出现0分，所以我又筛选出了考试0分的值来进行分析：

```
df.loc[(df['数学']==0)]
```

	三门总分	五门总分	准考证号	加一调整分	姓名	学校	学科	总分排名	数学	英语	语文
1532	92.0	209.2	11030241045	102.0	[illegible]	上大附中	物理	2085	0.0	92.0	0.0
2466	97.5	191.7	11080161030	81.0	[illegible]	通河中学	生物	2088	0.0	97.5	0.0

```
df.loc[(df['语文']==0)]
```

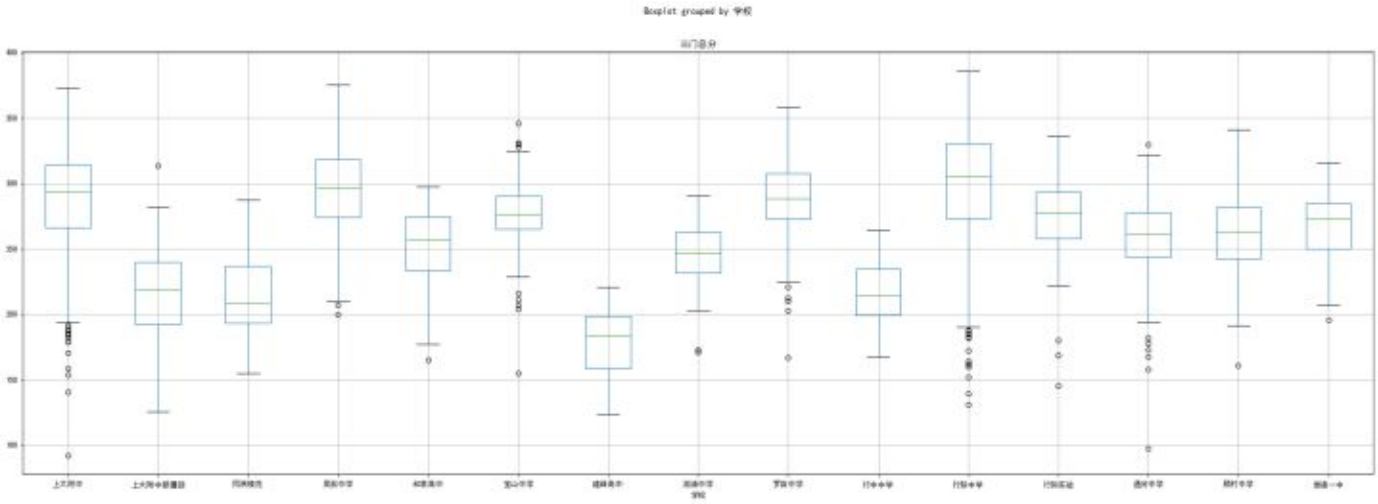
	三门总分	五门总分	准考证号	加一调整分	姓名	学校	学科	总分排名	数学	英语	语文
1532	92.0	209.2	11030241045	102.0	[illegible]	上大附中	物理	2085	0.0	92.0	0.0
2466	97.5	191.7	11080161030	81.0	[illegible]	通河中学	生物	2088	0.0	97.5	0.0

0分的原因有可能有缺考、作弊等等，但是奇怪的是这两位同学均有英语成绩，数学语文都是0分（这都说好的吗？？？）。而且这两位同学不在同一个学校，准考证也相去甚远，排除了作弊的可能性，我也是对这组数据非常不解

***另，在训练模型时，我们应该筛选出这种不符合常识的异常值单独讨论，因为线性回归模型在训练时非常容易受到异常值和噪音的影响。当然在这次建模中，我们不考虑这个问题，仅仅呈现最基础的步骤，方便理解。

之前我们主要在讨论平均数，但是事实上平均数并不能很全面地反应每一组的可观状况，因此这里我们引入**箱型图分析**：

```
df[['学校','三门总分']].boxplot(column='三门总分',by='学校', figsize=(30,10))
```



箱型图是可以观测数据分布的强大工具，但是原理非常简单易懂，建议自学

观察箱型块状我们发现：

1. 中间的箱型所处的位置越高，说明该学校的考生的平均水平越高，名列前茅的有上大附中，吴淞中学，罗店中学，行知中学（他们的确都是重点中学）

2. 行知中学的长方形面积最大，说明考生的成绩分布更为分散也说明成绩差距越大

观察异常值我们发现：

1. 上大附中与行知中学下限处有大量的异常值，说明有大量吊车尾的学生

a. 行知中学有一个“正式生转借读生”机制，对于成绩不好的学生，学校会将他的学籍转为借读生（一般挂在行中中学名下），从而不影响升学率。

b. 行知中学的这些异常值学生可能就会在高考前成为借读生

c. 上大附中可能存在同样的机制？

d. *** 如果尝试把行知中学中的这些异常值，加入行中中学的数据中，重新制作行中中学的箱型图，不知道还会不会有异常值?

2. 处于上限的异常值并不多，尽在宝山中学、上大附中新疆部、通河中学中存在，这些处于上限的学生可以理解为远高于该校平均水平的学生。很好奇这些学生的高考去向以及目前的状况如何，很希望有机会认识他们一下，我觉得将会是帮助我们理解统计学、理解人生非常有价值的数据。

三，用机器学习预测

终于到了激动人心的机器学习环节，好比一项赛事中对垒双方起初摩拳擦掌，互相试探，现在终于要进入到高潮了。诶嘿嘿，想想还有点小激动呢/(/ /•/ω/•/ /)/

坦白来讲，使用机器学习真的没有这么难！但是理解和吃透机器学习是非常难的！就如同我们谁都会用 1+1=2，但谁能证明出 1+1=2呢？（著名的哥德巴赫猜想）

所以不要复杂被理论吓退了，所以只要你会敲两行代码，你就一定会使用机器学习模型，就如吴恩达老师所说的，完成比完美更重要，让行动先于思考。

（当然只会代码不懂数学和统计，被称之为机器学习的danger zone，知行合一才是最好的学习方法）

好了回到主题，在做机器学习之前我们先明确目的（不是应该在做任何事之前都这样吗？？？），先看看我们有什么，以及我们要什么。

	语	数	英	加一调整分	综合	三门总分	五门总分
理科考生	√	√	√	√	√	√	√
文科考生	√	√	√	×	×	√	√

可见，我们理科生考生数据是完善的，但是文科生考生缺少加一调整分和综合分，这也正是我们要通过机器学习来进行预测的。实际上，我们只要预测其中一门的成绩即可，

剩余一门成绩= 五门总分 - 已知四门成绩 （这tm不是废话吗(ノ ‘□’)ノ ┻┻）

在这里，我们选择“综合”作为预测对象。

***（当然你也可以选择“加一调整分”作为预测对象，但是考虑物理、化学、生物、历史、政治、地理等“加一”学科的考卷差别非常大，很难找到共性，而“综合”学科的考卷是相同的，更容易找到共性，选择“综合”作为预测对象是出于对实际情况的考虑和对数据的理解）

那我们怎么预测呢？在预测之前，我们首先要对一种假设达成共识，该假设就是：

一个考生的“综合”成绩是与其他数据特征息息相关的。

若更具体地举例说明就是：

可能性假设1 - **如果一个学生语数英成绩都很好，是一个典型的优等生，那么一般来说他的综合成绩也不会太差**

可能性假设2 - **如果一群学生都是由同一个综合老师教出来的，那么这群学生接受的知识是差不多的，考出来的综合成绩也是趋同的，或者满足同一个正态分布**

可能性假设3 - **综合成绩与五门总分和三门总分的差相关，如果差越大，那么综合成绩可能越高（这句话对人类来说有点废话，但对机器来说，可以提供一个有效的特征）**

可能性假设4 - **如果准考证号接近的考生会被分配在同一个考场，那么将会受到相同的环境影响，例如温度、湿度、教师环境会影响考生的心情，例如突然事件例如有考生晕倒影响考场秩序，再例如同考场作弊等因素。**（突然想到可以分析准考证相近的考生的成绩离差，从而判断是否可能作弊）

可能性假设5 - ……

可能性假设6 - ……

如果你认同上述这些假设（写作假设，读作naodong），我们就可以愉快地进行下一步的分析。我们的这些假（nao）设（dong），可以告诉我们哪些数据可能是相关的，但是相关到什么程度，我们无法用肉眼或经验判别，就让机器告诉我们吧！

第一步还是简单的数据清洗：

```
#把df_1重命名为df_train
#并在df_train把计算出来的综合分加进去
df_train = df_1
zonghe = df_train.五门总分 - df_train.加一调整分 - df_train.数学 - df_train.英语 - df_train
df_train.insert(0,'综合',zonghe)

#查看一下各数据与综合成绩的相关性
df_train.corr()
```

	综合	三门总分	五门总分	准考证号	加一调整分	总分排名	数学	英语	语文
综合	1.000000	0.492785	0.562885	-0.184843	0.503469	-0.549845	0.446697	0.393162	0.323930
三门总分	0.492785	1.000000	0.970789	-0.499649	0.687717	-0.935707	0.865287	0.845612	0.661752
五门总分	0.562885	0.970789	1.000000	-0.482902	0.839779	-0.960022	0.859629	0.813141	0.612439
准考证号	-0.184843	-0.499649	-0.482902	1.000000	-0.345282	0.476039	-0.416798	-0.426240	-0.358904
加一调整分	0.503469	0.687717	0.839779	-0.345282	1.000000	-0.796146	0.654921	0.559998	0.359620
总分排名	-0.549845	-0.935707	-0.960022	0.476039	-0.796146	1.000000	-0.837761	-0.779738	-0.576954
数学	0.446697	0.865287	0.859629	-0.416798	0.654921	-0.837761	1.000000	0.528408	0.381667
英语	0.393162	0.845612	0.813141	-0.426240	0.559998	-0.779738	0.528408	1.000000	0.489917
语文	0.323930	0.661752	0.612439	-0.358904	0.359620	-0.576954	0.381667	0.489917	1.000000

很遗憾，没有找到特别相关的数据，但是我们依旧继续进度，尝试做出一个baseline模型先，我们打算把‘三门总分’，‘五门总分’，‘数学’，‘英语’，‘语文’这五组数据作为特征保留下来。

由于‘学校’特征是文字，无法用通过数值形式表现出来，在这里我们把‘学校’特征进行one-hot编码，方便用来给模型训练。

```
dummies_xuexiao = pd.get_dummies(df_train['学校'], prefix ='学校')
dummies_xuexiao.head()
```

学校_上大附中	学校_上大附中新疆部	学校_吴淞中学	学校_和衷高中	学校_宝山中学	学校_建峰高中	学校_淞浦中学	学校_罗店中学	学校_行中中学	学校_行知中学	学校_行知实验	学校_通河中学	学校_顾村中学	学校_高境一中
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0

这是进行one-hot编码后的结果，所在学校为1，其他为0，一般遇到类型数据，我们都会这样进行操作，以方便机器理解

把所有要用到的数据都合并到一个DataFrame中，并把用不到的数据都删除：

```
data_train = pd.concat([df_train, dummies_xuexiao], axis=1)
data_train.drop(['准考证号','姓名','学科','学校','加一调整分','总分排名'],axis=1,inplace=Tru
data_train.head()
```

	综合	三门总分	五门总分	数学	英语	语文	学校_上大附中	学校_上大附中新疃部	学校_吴淞中学	学校_和衷高中	学校_宝山中学	学校_建峰高中	学校_淞浦中学	学校_罗店中学	学校_行中中学	学校_行知中学	学校_行知实验	学校_通河中学	学校_顾村中学	学校_高境一中
731	18.4	359.5	494.9	127.0	123.5	109.0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
732	19.8	330.5	461.3	103.5	128.5	98.5	0	0	0	0	0	0	0	0	0	1	0	0	0	0
733	20.0	360.0	507.0	135.0	123.0	102.0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
734	19.4	360.5	521.9	145.0	124.5	91.0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
735	22.4	353.5	519.9	136.5	130.0	87.0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

要喂给模型的数据都在这里了

在训练模型之前，我们对已经结构化好的数据集做一次切分，只拿其中一部分去喂给模型，剩下的一部用来给模型测试准确率：

```
from sklearn.cross_validation import train_test_split

X = data_train.as_matrix()[: , 1:] #把DataFrame转换成ndarray，并且进行切片操作，X是数据特征，
y = data_train.as_matrix()[:, 0]   #同样操作，y是标签，也就是“综合”分数

X_train, X_test, y_train, y_test = train_test_split(X,y) #切分成训练集和测试集
```

好了，现在要训练模型了，终于到了激动人心的时刻！但事实上训练模型并不难，大部分的工作sklearn库都帮你预置好了，你只要调用模型即可，这里我们选择最简单的线性回归模型/LinearRegression:

```
from sklearn.linear_model import LinearRegression
model = LinearRegression() #建立模型
model.fit(X_train, y_train) #训练模型
```

好了，我们的模型在调用.fit以后就新鲜出炉了。事实上我们跳过了许多重要步骤例如选择模型、调参等等，但这不影响我们完成模型，只是影响我们模型的质量。

紧接着我们把测试集数据特征放进模型里，模型会告诉我们一组它自己的预测：

```
y_pred = model.predict(X_test)
y_pred
```

```
array([ 13.7278485 ,  14.655934  ,  15.69195273,  14.1362282 ,
        15.96628388,  16.58460227,  15.9465985 ,  13.53880534,
        15.68414023,  14.44612222,  15.29075847,  12.15623117,
        16.4634371 ,  13.30241837,  15.69700847,  15.65296165,
        15.56003388,  15.52892507,  15.00554337,  14.3269816 ,
        14.57200847,  14.85710587,  14.48278757,  14.51732097,
        16.14703085,  16.69479951,  14.2644816 ,  16.14220632,
        15.99859335,  17.2410441 ,  14.3931246 ,  19.06028972,
        16.17161784,  13.7566691 ,  14.6196871 ,  13.0872235 ,
        13.33331725,  13.98694602,  16.65403972,  16.02490008,
        14.51732097,  17.57671835,  13.7363093 ,  15.35911784,
        14.2722941 ,  14.68503388,  14.03288712,  13.76617148,
        14.8081032 ,  13.76614293,  15.89818034,  14.8347941 ,
        16.44390585,  14.18132462,  15.27878388,  17.63605163,
        14.28685222,  11.79443133,  13.60015585,  13.51513445,
```

模型根据输入的特征，给出了预测。emmmm.....综合分满分30份，乍眼看去，这组数据15分上下还挺合理的吗。我们进一步把之前预留的测试集数据的真实答案与模型给出的数据进行对比，来看看这个预测靠谱不靠谱。

（这里我们不讨论learning curve，CV等，我们只是简单对比一下真实综合分数与预测综合分数之间的异同）

```
#我们简单地把两组数据重构成DataFrame来进行观察
y1 = y_pred.reshape(-1,1)
y2 = y_test.reshape(-1,1)
var1 = pd.DataFrame(y1)
var2 = pd.DataFrame(y2)
var3 = var1 - var2
var = pd.concat([var1,var2,var3],axis=1)
var.columns =['预测分','实际分','误差']
var.head()
```

预测分	实际分	误差
13.727849	15.2	-1.472151
14.655934	14.8	-0.144066
15.691953	13.6	2.091953
14.136228	17.0	-2.863772
15.966284	18.0	-2.033716

乍看之下，误差好像还可以嘛！

```
var.describe()
```

	预测分	实际分	误差
count	523.000000	523.000000	523.000000
mean	14.980273	15.062715	-0.082442
std	1.462841	2.144307	1.630042
min	9.844296	8.400000	-5.010000
25%	14.023139	13.600000	-1.224809
50%	14.914044	15.000000	-0.073018
75%	15.888278	16.600000	1.060579
max	19.346250	22.200000	5.136309

1. 误差的平均值是 -0.08，看着还行

2. 误差的标准差是 1.6, 好像有点高，不过似乎也能接受

3. 误差的最大值和最小值是 ±5 ，好像不是很大

注意！！！上述三句话，都是**耍流氓，耍流氓，耍流氓**!

任何脱离数据环境所做出的的判断都是**耍流氓！！！**

我们切不可以主观地判断数值大小，以这组数据来说由于综合总分才30分，误差±5分，占据了总分的16.7%，任何一分误差的上下，都会导致考生排名的巨大浮动，从而影响升学率，设想假如你就少了一分不能进清华，你会觉得这一分误差不重要吗？因此所有判断都要尊重数据所处的实际情况。

但是话说回来，如果我们只是拿这组数据练手，这个模型可以用吗?

答案是完全可以用的，这个模型的预测结果也在可接受范围之内。

当然！！！我们还可以做更多来改进模型：

1. 每一分综合会在多大程度上影响排名？（由于对选拔制考试来说，排名是关键，我们将研究每一分浮动对排名的影响，因此来探讨对预测误差的容忍程度）

2. 尝试将语数英成绩标准化计分以后，是否对提高准确率有帮助?

3. 尝试寻找或挖掘出更多的特征，例如准考证号？姓名?

4. 为模型评分，改进模型，避免欠拟合和过拟合

5. 尝试用其他模型来做预测

当然，这些工作量也比较大，我们篇幅也挺长了，这些我们就下次在做（如果我还记得的话）

注意！！！我们的活还没干完哦，我们只建立了模型，还没给文科生的综合成绩做预测呢。emmmm.....但是需要重新结构化一遍文科生的数据，好吧，我承认我懒得做了，就交给聪明的你来做吧~

若想要数据集，请加我微信并发送“考试数据”：

微信号: keypaladin

再说一遍！！！！！！真的没有番号！！！！！！！！

编辑于 2018-04-15

数据分析　Python　机器学习