深入理解PCA降维

机器学习实验室 2020-04-01

以下文章来源于AI蜗牛车 ，作者Miracle8070

AI蜗牛车

一个慢慢爬的小蜗牛，蜗牛坐在车里，可以更快地往高处走~ 分享时间序列、时空序列...

机器学习

Author：Miracle8070

From：AI蜗牛车

1. 写在前面

如果想从事数据挖掘或者机器学习的工作，掌握常用的机器学习算法是非常有必要的，在这简单的先捋一捋， 常见的机器学习算法：

- 监督学习算法：逻辑回归，线性回归，决策树，朴素贝叶斯，K近邻，支持向量机，集成算法Adaboost等
- 无监督算法：聚类，**降维**，关联规则, PageRank等

为了详细的理解这些原理，曾经看过西瓜书，统计学习方法，机器学习实战等书，也听过一些机器学习的课程，但总感觉话语里比较深奥，读起来没有耐心，并且**理论到处有，而实战最重要.**

个人认为，理解算法背后的idea和使用，要比看懂它的数学推导更加重要。idea会让你有一个直观的感受，从而明白算法的合理性，数学推导只是将这种合理性用更加严谨的语言表达出来而已，打个比方，一个梨很甜，用数学的语言可以表述为糖分含量90%，但只有亲自咬一口，你才能真正感觉到这个梨有多甜，

也才能真正理解数学上的90%的糖分究竟是怎么样的。如果这些机器学习算法是个梨，本文的首要目的就是先带领大家咬一口。另外还有下面几个目的：

- 检验自己对算法的理解程度，对算法理论做一个小总结
- 能开心的学习这些算法的核心思想， 找到学习这些算法的兴趣，为深入的学习这些算法打一个基础。
- 每一节课的理论都会放一个实战案例，能够真正的做到学以致用，既可以锻炼编程能力，又可以加深算法理论的把握程度。
- 也想把之前所有的笔记和参考放在一块，方便以后查看时的方便。

学习算法的过程，获得的不应该只有算法理论，还应该有乐趣和解决实际问题的能力！

PCA（Principal Component Analysis）降维是一种无监督的方式，常用在数据的降维方面，也就是数据的预处理方面，**并不是作为最终的算法模型使用，只是相当于一种帮助算法更好更快做出决策的辅助手段**。因为我们知道，如果给我们的数据特征维度太高，首先计算很麻烦，其次特征之间可能存在相关的情况，从而增加了问题的复杂程度，分析起来也不方便。这时候我们就会想是不是去掉一些特征就好了呢？但是这个特征也不是凭自己的意愿去掉的，因为盲目减少数据的特征会损失掉数据包含的关键信息，容易产生错误的结论，对分析不利。

所以我们想找到一个合理的方式，**既可以减少我们需要分析的指标，而且尽可能多的保持原来数据的信息**，PCA就是这个合理的方式之一。所以今天我们学习PCA降维，看看到底PCA再完成一个什么样的任务，是如何完成这个任务的？

首先，我们先从生活的场景出发感受一下降维的身影，然后我们会从二维数据的降维例子看看降维在做什么事情，从宏观的层面介绍PCA在完成什么样的任务，然后我会从数学的微观层面上讲解PCA是如何完成降维任务的，最后我们会用纯Python实现PCA算法完成鸢尾花数据集的分类，接着会调用sklearn的pca工具来做一个人脸识别的降维分析，看看PCA到底在实战任务中是怎样的一个存在。 总之有了这个手段，你的模型分析起数据来会更加的得心应手。最后也要提醒一句，虽然这次也是白话系列，但是这次会有一些数学的东西在里面，毕竟没有数学，PCA再怎么描述也是灵魂出窍般的存在，但保证不会太难，并且有我的大白话在里面，轻轻松松感受数学的强大。哈哈，我们开始吧！对了，PCA也叫主成分分析，这俩是一回事！

大纲如下：

- 数据降维？ 其实就在我们身边！

- PCA到底在做什么？ （宏观的角度把握）
- PCA是如何做到降维的？ （微观角度理解PCA的数学原理）
- PCA编程实战（鸢尾花数据集+人脸识别的降维分析）

OK, let's go!

2. 数据降维？ 其实就在我们身边！

数据降维的身影，我们身边也会有很多，我们常常说**算法来源于生活**，只不过我们在人生路上走的太快，没有去留意路边的风景罢了。没事，只要你喜欢听听别人的故事，我替你留意了一些，哈哈。下面我们就从一个故事场景中感受降维：

> ★ 我们走在一个城市中，总是会发现城市中的每一条道路会有一些奇怪的名字，什么北京路，京哈路什么各种，我就拿我所在的城市来讲，道路起名字还算规范，因为我们这东北是渤海，南边是黄河，所以城市中横向的路统称为黄河，然后从最南边数，往北依次是黄河一路，黄河二路，黄河三路....， 纵向的路统称为渤海，从东边往西依次是渤海一路，渤海二路，渤海三路...,和我们坐标系的X轴和Y轴是非常像的。可能刚到某个城市的时候你会有个疑问，为啥会给路起名字呢？有的甚至起一些花里胡哨的名字，听都没听过。这是因为，给街道起名字之后，城市中的位置信息都可以通过街道名唯一确定了（比如黄河五路和渤海三路交叉口），这样当你想找某个地方的时候，才更容易找到，毕竟给路起名之后，是大家公认的标准，当大家都说黄河五路和渤海三路交叉口的时候，保证说的是同一个位置，这些路就类似于我们说的坐标系。而城市中的每一个地方就类似于坐标系中的点，被坐标系唯一标识。那么有个问题来了，我想让你把黄河五路渤海三路交叉口这样用两个数字标注的位置信息，转换为只有一个数字标识的位置描述，你怎么办？ 你可能没法一下子想到。但是如果有一条铁路通过了城市，而城市中所有重要的建筑都在铁路边，那么就可以根据距离铁路的起点多远来定义每一个点的位置。那样是不是这个问题就解决了？比如黄河五路渤海三路交叉口距离铁路挺近的，在起点旁边。 其实，**这就是一种降维了，把原来的需要两个维度标注的信息，现在用一条铁路就可以搞定** ”

当然，这样的定位不如用俩个维度来的准，有的地点离铁路远，但是远多少，在新的表示中就没有得到展示了。**这说明数据降维不是无损的，会造成信息的部分丢失。**

那么，降维有什么用呢？ 只是为了造成信息的部分丢失？还是回到上面的例子，数据降维的第一个用途是**数据压缩**，如果你只能在一张小便签向一位你新认识的朋友写下你家的地址，便签上写不下是黄河××路渤海××路，那你可写铁路第五。而数据降维还可以去做**数据可视化或特征提取**，比如你要在城市中开一家店，你先看看哪里人群更加密集，你可以通过数据降维，做出哪些地方周围的点多，从而人流更密。数据降维的第三个用途是**异常值检测和聚类**，例如你通过数据降维，发现城市中的大部分人家都住在两个火车站附近，但是有一俩个家却不在这里，这样你就发现了城市里那些特立独行的人，接着你发现俩个火车站附件的人家，一家都姓张，一家都姓李，这样，你就将城市的人家通过数据降维，分成了俩类。

现在对降维有了一个直观的认识了吧， 那么这个和我们的主角PCA有什么关系呢？主成分分析方法是一种数据降维的方式，刚才上面我们提到，只要数据压缩，必定会损失一些信息，而PCA做的就是尽可能的去找到一些主要的关键特征去区分开数据，去除掉一些对区分数据不大的那些特征，这样，既可以做到降维，也可以尽可能多的保留原来数据的信息。在上面的例子中，我们找到的铁路线就是我们的主成分，我们将城市的所有人家按照铁路开来的顺序，依次排序，从而得到了一个数字表示的距离，至于位置信息的损失，肯定是损失一些，但至少能表示。

下面我们再详细看看PCA究竟在做什么？

3. PCA到底在做什么？ （宏观的角度把握）

上面，我们已经通过生活的场景感受到了降维的身影，那么也知道了主成分分析方法是降维的一种方式，它能够尽可能多的保留数据的信息而完成减少数据的维度，上面我还说，那条铁路就是我们找出的主成分，但是在数据中主成分到底是什么样子呢？我们先通过一个例子来感受一下：下表是某些学生的语文、数学、物理、化学成绩统计：

表 1

学生编号	语文	数学	物理	化学
1	90	140	99	100
2	90	97	88	92
3	90	110	79	83
. . .	. . .	. . .	. . .	. . .

首先，假设这些科目成绩不相关，也就是说某一科目考多少分与其他科目没有关系。现在我想让你用尽量少的科目成绩来区分这三个同学学习等级的话，我想你

一定不会选语文成绩作为区分的标准（因为语文分数都一样啊，没有区别啊），你一眼就可以看出来数学、物理、化学可以作为这组数据的主成分（很显然，数学作为第一主成分，因为数学成绩拉的最大），很大程度上我们光用数学成绩就可以区分开这三个同学吧。这和我们平时考试一样，为啥父母都喜欢说好好学数学，因为数学才能真正的拉开分决定谁考第一名。 这不，单单一门数学成绩就可以完成分类任务了，不用考虑其他三科，这不就降维了啊。

那你怎么就确定数学就是主成分的呢？ 你说，因为数学拉开了分啊， 你这个拉开了分，其实就是说的这个分数在同学和同学之间的差距可能会比较大，这个正是我们概率论上常常讲的**方差**。方差越大，所获得的的信息量就会越多。**而PCA找主成分的时候其实在寻找K个尽可能的把样本区分开方向，即方差尽可能大的方向作为主成分，这样就可以做到在保留尽可能多的数据信息的情况下把数据的维度降到了K维（原来肯定是比K维大）**

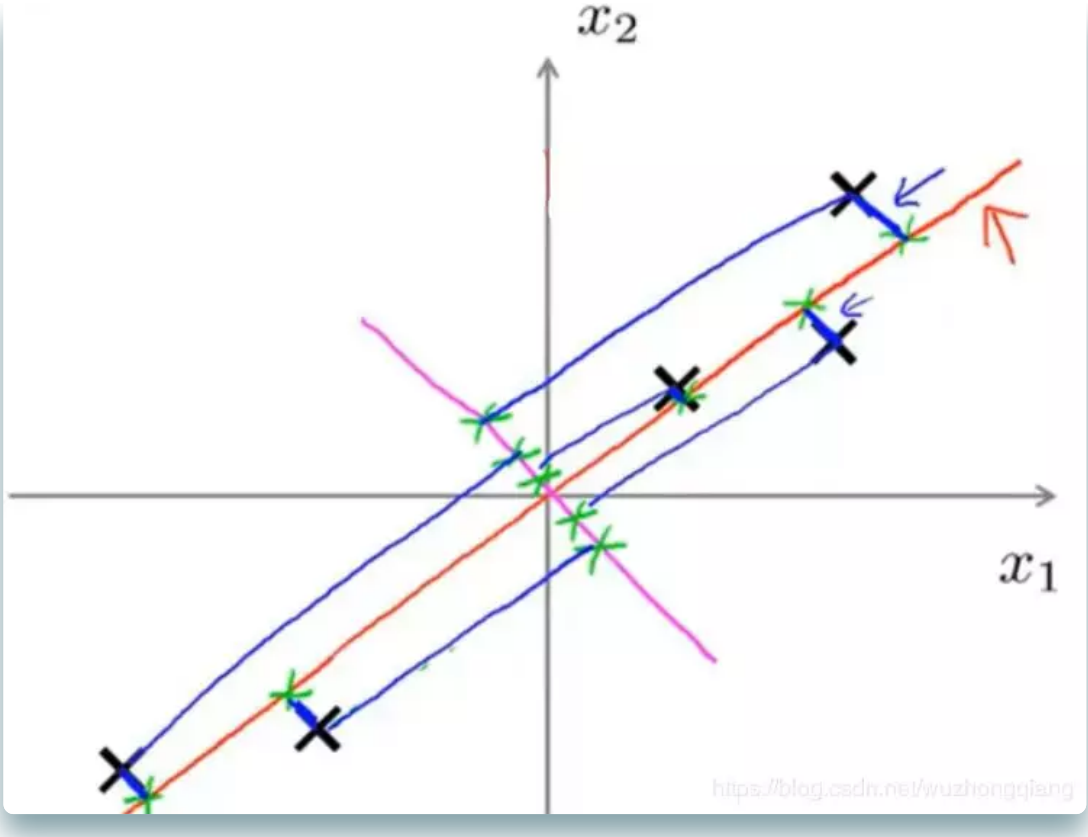
好了，那么上图比较简单，你一下子就能看出主成分了，我们再看一组数据分数：

表 2

学生编号	数学	物理	化学	语文	历史	英语
1	65	61	72	84	81	79
2	77	77	76	64	70	55
3	67	63	49	65	67	57
4	80	69	75	74	74	63
5	74	70	80	84	82	74
6	78	84	75	62	72	64
7	66	71	67	52	65	57
8	77	71	57	72	86	71
9	83	100	79	41	67	50
. . .	. . .	. . .	. . .	. . .	. . .	. . .

你还能一下子找出哪一科可以作为主成分吗？ 你可能又会说，这还不简单，这还不简单？你不是说了找方差极可能大的吗？我算一算每一科的方差，然后我看看哪几个方差最大不就行了？哈哈，这个地方注意不要理解错了，寻找主成分并不一定是在这原有的这几科上去找某几科方差最大的科目，而更像是这些科目的一种均衡，这啥意思？就比如，我们的每个科目都代表一个方向，那么每一个学生在空间中根据每门分数不同就成了一个个的点，我们想要找的主成分，不一定是这原来的某个方向或者某些方向，我们找的方向，是这每个学生投影上来之后能够离得尽可能的远，也就是方差尽可能大，这样有利于我们的区分。

还是拿吴恩达老师的这个图来看一下：



我们看二维里面的这几个数据点， 如果我们找那条橙色的线划分这几个样本的话，你会发现每个点在橙色线上的投影离得距离很大，这样即使用这一条线划分数据的话，依然很容易划分出来。但是如果找那条洋红色的线划分数据的话，会发现每个点在线上的投影离得很近，不太好区分开。所以我们更倾向于选择橙色的线作为主成分。 ”

所以，对于上面那个多维的那些分数，我们更倾向于找一个下面这样的轴来分开数据（换一个角度思考问题）：

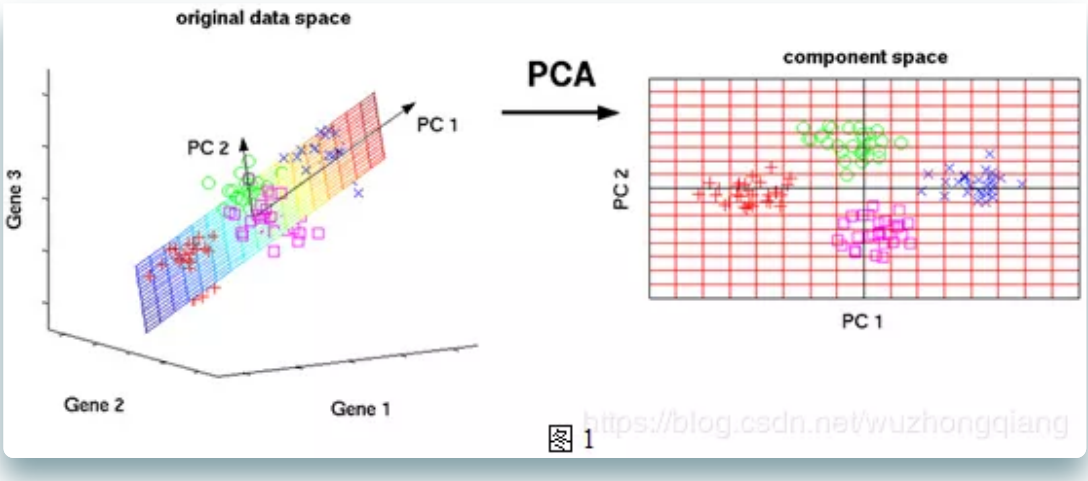


图 1

嗯嗯，在宏观上理解PCA的话，上面就是PCA做的事情 --- 寻找主成分，尽可能的去把数据分开而又不太损失原来的信息。

具体的说，就是PCA在降维的时候， 是把数据看成空间中的点，然后尝试去寻找几个方向（上面橙色的和下面的PC1PC2）， 把这些点进行投影，投影之后让这些点离得尽可能的远。这样这几个方向就是主成分， 空间中的样本点就可以通过这几个新的方向进行描述了。 但是找的这几个方向也是有要求的，就是互不干扰，没有线性关系，就像x轴和y轴那样，这样才能更好的去描述这些数据。主成分之间没有冗余。

所以，主成分的标准两个条件，一是互不相关，二是用来描述数据的时候，方差尽可能大。

那么PCA是怎么做到的呢？ 这个就得从数学的角度进行理解了。

4. PCA是如何做到降维的？（微观角度理解PCA的数学原理）

谈到数学这一块，就得严谨一些了,我们上面知道了PCA就是去找主成分，而主成分的标准两个条件，一是互不相关（注意不相关可不等于互相独立，这里只保证没有线性关系）；二是用来描述数据的时候，方差尽可能大，也就是数据投影过去之后，离得尽可能远。

接下来，其实就是弄明白PCA是怎么去衡量这两个条件的，第一个条件的话，互不相关，可以找一组主成分使得彼此之间的协方差为0（后面会提到）， 那么第二个条件，数据投影过去之后，离得尽可能远？ 这个有点麻烦，所以看第二个之前，我们先来看看什么是投影，这个怎么衡量？

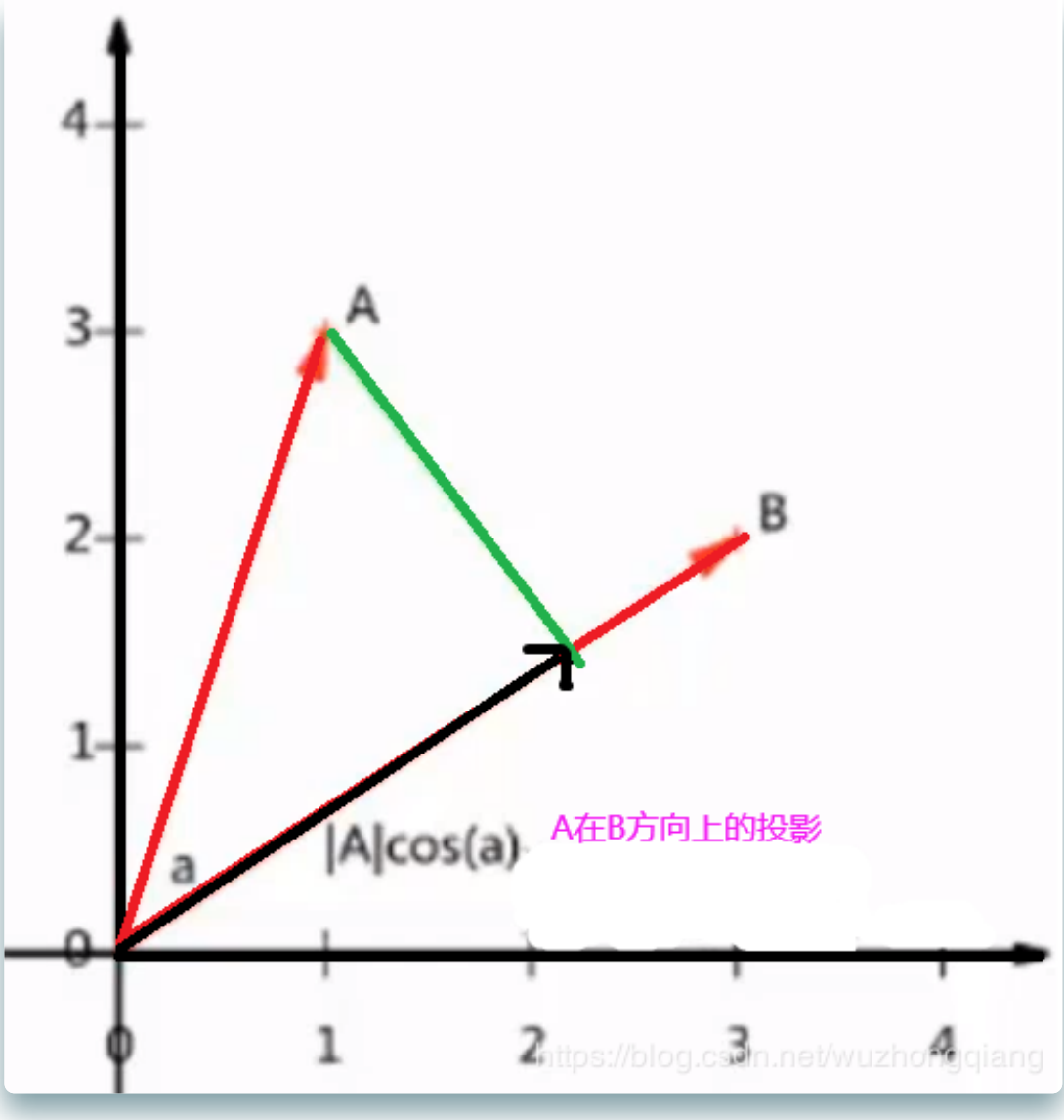
这个就得先从向量的表示和基变换说起了：

4.1 向量的表示及基变换

要想看投影，我们先来看看向量的内积运算是怎么回事来？

内积： $(a_1, a_2, \ldots, a_n)^T \cdot (b_1, b_2, \ldots, b_n)^T = a_1b_1 + a_2b_2 + \ldots + a_nb_n$ **解释：** $A \cdot B = |A||B|cos(\alpha)$

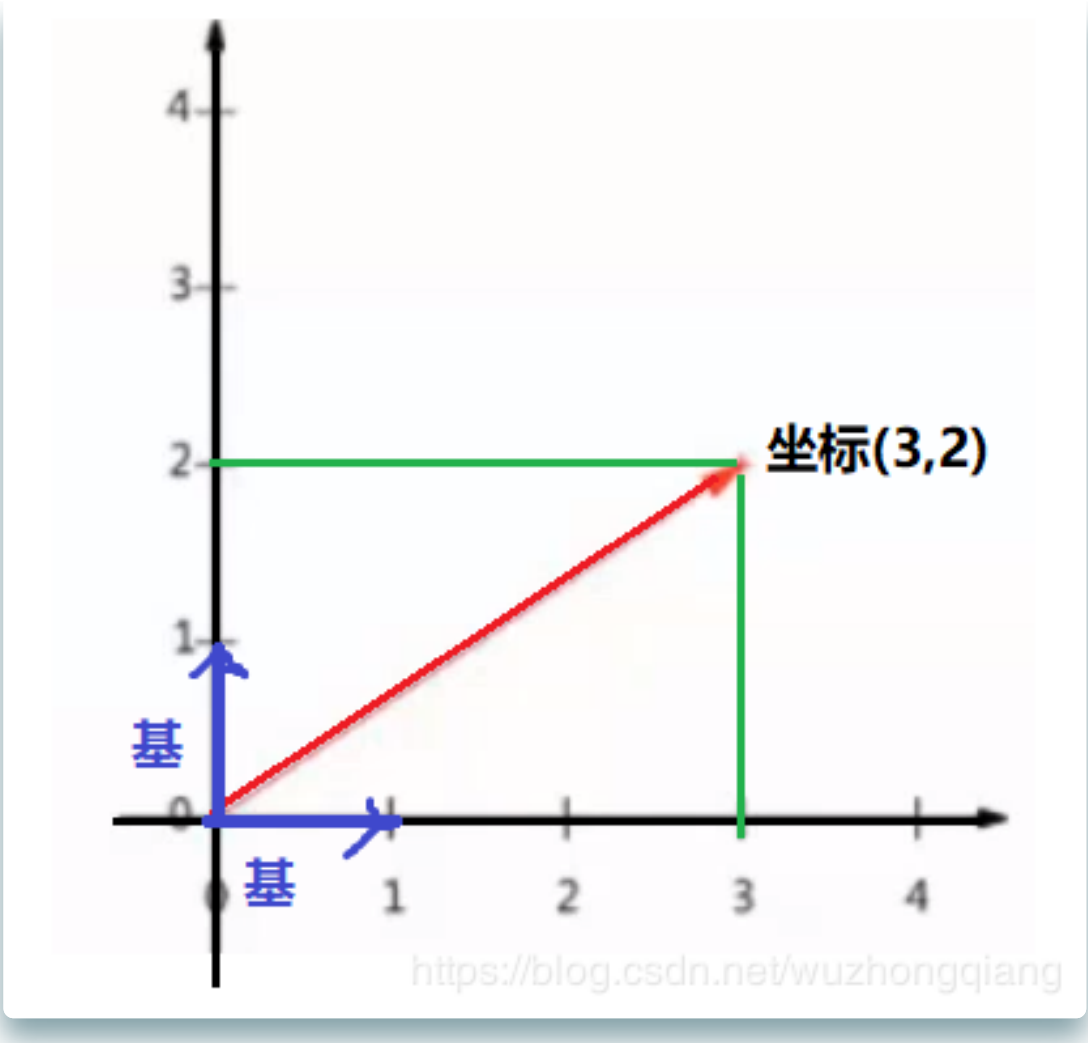
那么，如果向量B的模为1， 那么**A与B的内积值等于A向B所在的直线进行投影的矢量长度**。看下面的图片：



所以，两个向量进行内积之后，相当于做了一个投影的操作。

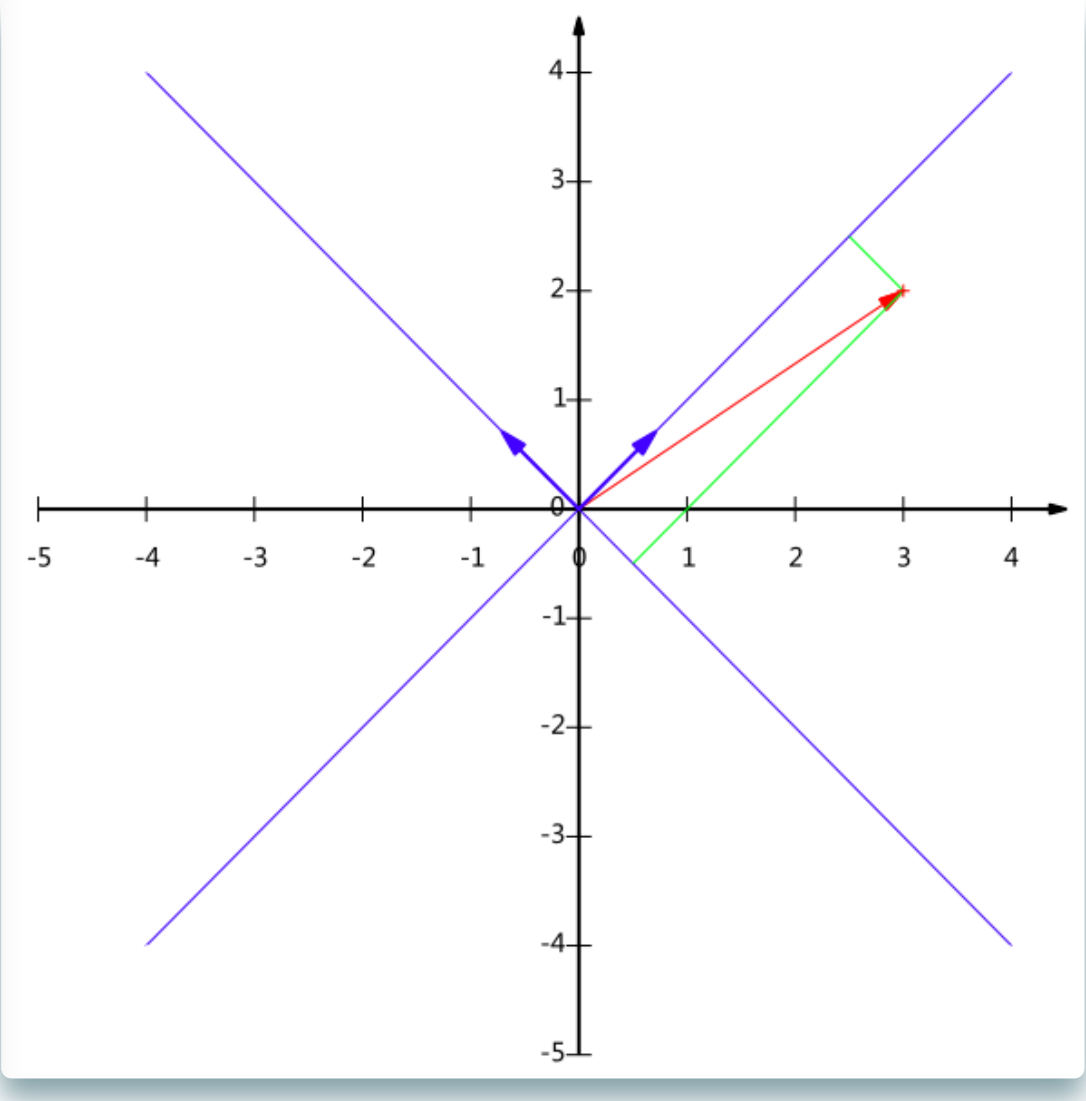
那么，我们看看向量在空间中是如何表示的：

我们现在空间中选定一组基，那么空间中的向量都可以表示为基的线性组合，这是什么意思呢？比如我们下面确定了一个坐标系，那么下面这个红色的向量，坐标是(3,2)，就是因为这个坐标在x轴上的投影是3， 在y轴上的投影是2，实际上，这个向量可以表示成线性组合：$x(1,0)^T + y(0,1)^T$



而上面的(1,0)和(0,1)就叫做二维空间中的一组基。并且要求基是单位向量且要垂直，不相关，如果不知道讲基干啥用，那么还记得主成分的第一个条件吗？也是互不相关，难道只是巧合？NO，只不过这里得先做一些铺垫，有点想要呼之欲出的感觉。

我们学这个基到底干啥用呢？ 先别慌，有了基之后，我们就可以做基变换了，啥意思？就是说，上图中我们知道了红色向量的坐标是（3,2），但这样说不准确，正确的应该是先有了坐标系，在我这个x轴和y轴的坐标系下，我的红色向量坐标(3,2)。那么，我要是换一组坐标系呢？看下图，我问你，在蓝色的坐标系中，红色向量的坐标是什么？很显然，不是(3,2)了吧，所以以后考虑问题的时候，要考虑的全面一些，万事成立都是基于某些条件的，哈哈，没想到基变换中还能学到人生哲理。



那么，换了这个坐标系之后，我们原来的(3,2)到底变成了多少呢？我们依然是需要去找新坐标系中的基，看上面的蓝色箭头，就会发现这组基可以是$(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}})$和$(-\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}})$，找到基之后怎么做变换呢？

变换：数据与第一个基做内积运算，结果作为第一个新的坐标分量，然后与第二个基做内积运算，结果作为第二个新坐标的分量。数据(3,2)映射到新基中的坐标：

$$\begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{pmatrix} 5/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}$$

所以，在蓝色的坐标系中，红色的向量坐标为$(5/\sqrt{2},-1/\sqrt{2})$

那么，明白了上面的变换之后，我们就可以得到基变换的更为一般的表达方式：

$$\begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_R \end{pmatrix} (a_2 \quad \cdots \quad a_M) = \begin{pmatrix} p_1a_1 & p_1a_2 & \cdots & p_1a_M \\ p_2a_1 & p_2a_2 & \cdots & p_2a_M \\ \vdots & \vdots & \ddots & \vdots \\ p_Ra_1 & p_Ra_2 & \cdots & p_Ra_M \end{pmatrix}$$

两个矩阵相乘的意义是将右边矩阵中的每一列列向量变换到左边矩阵中的每一行行向量为基所表示的空间中去。其中a_i是列向量，p_i是行向量，一个基。理解起来就是左边的矩阵是一组基，右边的矩阵是样本集。这两个矩阵相乘，就是样本集在一个由一组新的基定义的空间中的表示。

这个基和我们的PCA有什么关系呢？ 现在就可以说一说了：

PCA做的就是寻找到一组基（主成分），这组基互不相关，并且使得所有的数据变换为这组基上的坐标表示之后，方差值尽可能的大。

我们知道一组基是互不相关的，但是PCA在具体选的时候，应该怎么选呢？PCA肯定不是一次就把所有方向都选出来的， PCA其实是这么做的，先选择一个方向基，让数据投影到这个基上的方差最大，然后再从和这个方向基正交的方向上选择第二个方向基使得数据投影到这个基上的方差最大，然后再从与这两个方向基都正交的方向上选择第三个方向基，这样依次选择下去，就可以保证方向基之间是互相正交的，也可以保证方差最大了。那么谈到方向基互相正交，就避不开数学上的另一个概念了：协方差。先来看看这个概念：

4.2 协方差

如果说方差表示的一个变量之间数据之间的波动程度，那么协方差表示的两个变量之间的相关程度， 两个变量X和Y的协方差可以表示成：

$$\operatorname{cov}(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{n-1}$$

如果我们把X和Y的均值变成0的时候（往往我们会先把数据做归一化处理），这时候，协方差就变成了

$$\operatorname{cov}(X,Y) = \frac{\sum_{i=1}^{n} X_i Y_i}{n-1}$$

所以，如果想让两个基没有相关性，就要保证他们的协方差为0.

4.3 PCA的优化目标

谈完了基变换和协方差，下面我们就很容易把PCA的那两个条件给用数学的语言表达一下了：

> ★ 将一组N为向量降维K维（0<K<N），目标是选择K个单位正交基，使得原始数据变换到这组基上后，各字段两两间协方差为0，字段的方差则尽可能大
> 那么怎么得到两个基的协方差信息呢？我们需要一个协方差矩阵的东西。 ”

我们假设我们的数据集X是2*m的一个矩阵，2表示两维的特征，m表示m个样本，那么，我进行X内积操作看看得到了啥：

$$X=\begin{pmatrix}a_1 & a_2 & \cdots & a_m\\ b_1 & b_2 & \cdots & b_m\end{pmatrix} \quad \frac{1}{m}XX^{\top}=\begin{pmatrix}\frac{1}{m}\sum_{i=1}^{m}a_i^2 & \frac{1}{m}\sum_{i=1}^{m}a_ib_i\\ \frac{1}{m}\sum_{i=1}^{mi}a_ib_i & \frac{1}{m}\sum_{i=1}^{m}b_i^2\end{pmatrix}$$

发现了吗？得到的后面的这个矩阵就是特征的协方差矩阵了，这个矩阵主对角线上的两个元素正好是特征自身的方差，而副对角线上的两个元素正好两个特征之间的协方差啊，是不是很神奇！这样一下子，特征自己的方差和特征之间的协方差直接就全了。并且这个协方差矩阵是个对称矩阵哟！

那下面的任务就明确了，PCA是干啥来，那两个标准，一个就是需要特征之间的协方差为0，怎么做呢？就是让**协方差矩阵相似对角化**， 另一个就是尽量的使得方差最大，就是**协方差矩阵对角化之后，把对角线上的元素从大到小排列**。

> ★ 协方差矩阵对角化：即通过变换让协方差矩阵变成除对角线外的其它元素为0， 并且对角线上的元素按从大到小的顺序排列
>
> $$PCP^{\top}=\Lambda=\begin{pmatrix}\lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n\end{pmatrix}$$
>
> ”

那么我们怎么把这个协方差相似对角化呢？ 这就涉及到线代的知识了，我们首先有这么个定理：

> ★ 实对称矩阵：一个n行m列的实对称矩阵一定可以找到n个单位的正交特征向量$E=(e_1,e_2,...e_n)$使得
>
> $$E^\top CE=\Lambda=\begin{pmatrix}\lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n\end{pmatrix}$$
>
> "

且λ_i是协方差矩阵C的n个特征值。

这样，**我们只要把特征值从大到小，对应的特征向量从上到下排列，则用前K行组成的矩阵乘以原始矩阵X，就得到了我们需要的降维后的数据矩阵Y**。也会是PCA之后的方式，而前K个特征向量组成的就是新空间下的一组基，也是PCA找的K个主成分。讲到这里你可能会有点懵，为啥呢？ 为啥协方差矩阵的特征向量把协方差矩阵相似对角化，并把特征值从大到小排列之后，对应的特征向量就是那组基呢？这里不太明白了，对吧。

那么好，我们带着这个疑问往下走一步，就是假设这个对应的特征向量真是那组基，那么我们用这个特征向量乘以原始矩阵X就会得到降维后的数据矩阵Y。即$Y={E_{e_1:e_k}}^TX$那我们来看看，这个降维后的数据矩阵Y自己做内积然后乘以1/m是什么东西，即

$$\frac{1}{m}YY^T=\frac{1}{m}({E_{e_1:e_k}}^TX)(X^TE_{e_1:e_k})={E_{e_1:e_k}}^T\frac{1}{m}XX^TE_{e_1:e_k}={E_{e_1:e_k}}^TCE_{e_1:e_k}=$$

这样发现什么了？竟然这个降维后的数据矩阵Y与自己内积得到的是新的维度下的协方差矩阵，而这个协方差矩阵正好是之前X的协方差矩阵C相似对角化后取得前K个特征值组成的对角阵。

这就说明，PCA找到的主成分之间互不相关，且主成分本身的方差是从N维中选的前K个最大的值。正好符合我们之前的目标。所以这个特征向量作为新的基对

X进行变换成Y的过程是完全合理的。

这就是PCA在数学的角度是如何找到那K个主成分的详细过程了。

先梳理一下这个过程，然后我们根据一个例子感受一下这个过程，遇到新的数据集X之后（N维），我们想降到K维，PCA是这样做的

- 首先，应该先把X每一维进行归一化，把均值变成0
- 然后，计算协方差矩阵C，即$C = \frac{1}{m} X_{norm} X_{norm}^T$
- 然后，计算C的特征值和特征向量，把特征值从大到小排列，对应的特征向量从上往下排列
- 然后，将协方差矩阵相似对角化
- 然后，去特征向量的前k个，得到K个基
- 最后，用这K个基与X相乘，得到降维后的K维矩阵Y

下面我们看一个例子，感受一下这个过程：

1. 输入数据X：2个特征，5个训练样本, 并去均值

$$X_{norm} = \begin{pmatrix} -1 & -1 & 0 & 2 & 0 \\ -2 & 0 & 0 & 1 & 1 \end{pmatrix}$$

2. 计算协方差矩阵C，即$C = \frac{1}{m} X_{norm} X_{norm}^T$

$$C = \frac{1}{5} \begin{pmatrix} -1 & -1 & 0 & 2 & 0 \\ -2 & 0 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} -1 & -2 \\ -1 & 0 \\ 0 & 0 \\ 2 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{6}{5} & \frac{4}{5} \\ \frac{4}{5} & \frac{6}{5} \end{pmatrix}$$

3. 计算特征值和特征向量特征值：

$$\lambda_1 = 2, \lambda_2 = \frac{2}{5}$$

特征向量：

$$c_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, c_2 = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

特征向量单位化：

$$c_1 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}, c_2 = \begin{pmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}, E = \begin{pmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix}$$

4. 相似对角化

$$E^\top CE = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} \begin{pmatrix} 6/5 & 4/5 \\ 4/5 & 6/5 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 0 & 2/5 \end{pmatrix}$$

5. 降到1维，那么就取c_1作为基，得最后结果

$$Y = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} \begin{pmatrix} -1 & -1 & 0 & 2 & 0 \\ -2 & 0 & 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} -3/\sqrt{2} & -1/\sqrt{2} & 0 & 3/\sqrt{2} & -1/ \end{pmatrix}$$

到这里，关于PCA的数学原理就整理完了，你消化了吗？ 没消化的话，下面的代码部分仍然可以帮助你理解这个过程，下面我们就用鸢尾花数据集，来手写PCA对其降维。

5. PCA编程实战

关于PCA实战，我们已经知道了PCA一般是用于数据预处理，对数据进行降维，对后面的模型更好的工作起很强的辅助作用。下面我们先从一个简单的数据集鸢尾花，来看看如何将上面的数学过程写成Python代码的方式，并对鸢尾花数据集降维看看最后的效果。

5.1 鸢尾花数据集的降维分析

首先，我想用鸢尾花数据集（因为简单一些，好理解）按照上面的PCA的过程，具体实现一下，看看降维到底有个什么效果？那么我们开始吧：

回忆一遍上面的过程：

1. 得到数据，把数据归一化
2. 然后得到协方差矩阵C
3. 求协方差矩阵的特征值和特征向量，并将特征值排序
4. 取前K个特征向量作为基，最后与X相乘得到降维之后的数据矩阵Y

下面就来模拟一遍

1. 先导入包（鸢尾花数据集在sklearn.datasets里面）

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

from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
```

2. 导入数据，并把它进行归一化

```
iris = load_iris()
X = iris.data   # Xshape(150, 4)

# X的归一化
X_norm = StandardScaler().fit_transform(X)
X_norm.mean(axis=0)      # 这样每一维均值为0了
```

3. 下面就是用PCA进行降维的过程

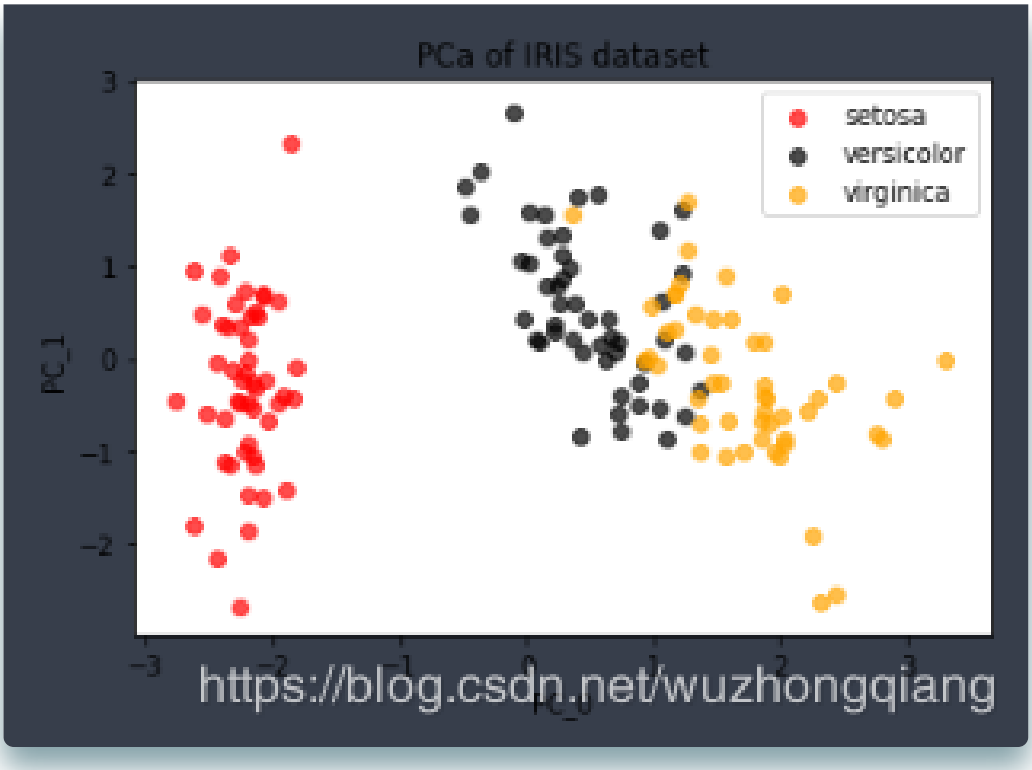
```
# 求特征值和特征向量
ew, ev = np.linalg.eig(np.cov(X_norm.T))   # np.cov直接求协方差矩阵，每一行代表一个特征，每一轮
```

```
# 特征向量特征值的排序
ew_oreder = np.argsort(ew)[::-1]
ew_sort = ew[ew_oreder]
ev_sort = ev[:, ew_oreder]  # ev的每一列代表一个特征向量
ev_sort.shape # (4,4)

# 我们指定降成2维， 然后取出排序后的特征向量的前两列就是基
K = 2
V = ev_sort[:, :2]  # 4*2

# 最后，我们得到降维后的数据
X_new = X_norm.dot(V)    # shape (150,2)
```

4. 下面我们可视化一下X_new，看看降维之后是什么样子：

```
colors = ['red', 'black', 'orange']

plt.figure()
for i in [0, 1, 2]:
    plt.scatter(X_new[iris.target==i, 0],
                X_new[iris.target==i, 1],
                alpha=.7,
                c=colors[i],
                label=iris.target_names[i]
               )

plt.legend()
plt.title('PCa of IRIS dataset')
plt.xlabel('PC_0')
plt.ylabel('PC_1')
plt.show()
```

结果如下：



从结果中，我们可以看到，PCA降维之后，首先是特征变成了2列，变得能够可视化出来，然后发现鸢尾花数据的每一类其实是比较容易区分开的，所以后面用一些基础的机器学习算法比如决策树，KNN等这些都可以得到比较好的效果，这样不仅可是使得计算简单，也为我们后面选择算法提供一个依据。

上面编写这个过程是为了更好的消化数学公式，其实sklearn中已经帮我们完成了这个工具， 我们直接可以用，并且完成降维只需一句话就可搞定，是不是简单多了。下面我们也可以试试：

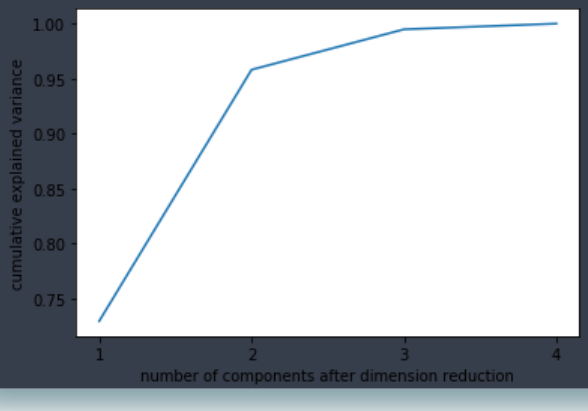
```
from sklearn.decomposition import PCA

# 然后使用
pca = PCA(n_compoents=2)
X_new = pca.fit_transform(X_norm)

"""查看PCA的一些属性"""
print(pca.explained_variance_)     # 属性可以查看降维后的每个特征向量上所带的信息量大小（可解释性方差的
print(pca.explained_variance_ratio_)  # 查看降维后的每个新特征的信息量占原始数据总信息量的百分比
print(pca.explained_variance_ratio_.sum())     # 降维后信息保留量

## 结果：
[4.228241710.24267075]   # 可以发现，降维后特征的方差
[0.924618720.05306648]  # 降维后的特征带的原有数据的信息量的比例
0.977685206318795# 降维后的信息保留（损失了3%， 去掉了一半特征，还算可以）
```

这样就实现了X_norm的降维操作，关于sklearn里面的PCA，详细细节可以去看参考文档，这里再简单的说一下PCA的几个小细节：

- pca的属性我们可以通过pca.explained_variance_ 属性， 查看降维之后每隔特征向量所带的信息量大小；pca.explained_variance_ratio_查看降维后的每个新特征的信息量占原始数据总信息量的百分比；pca.explained_variance_ratio_.sum()，降维后信息保留量
- 参数n_coments怎么选择？这个参数可以直接指定我们要降到几维（一个整数，大于0小于数据总维数）,当然这一个可能对于陌生的数据我们不太确定一下子是几维，可以画个图探索一下：

```
"""怎么选择n_coments呢？  画图，选转折点"""
pca_line = PCA().fit(X_norm)
plt.plot([1, 2, 3, 4], np.cumsum(pca_line.explained_variance_ratio_))
plt.xticks([1, 2, 3, 4])
plt.xlabel('number of components after dimension reduction')
plt.ylabel('cumulative explained variance ratio')
plt.show()
```

executed in 111ms, finished 22:31:33 2020-03-02



n_coments = 'mle'，这是自选超参数， 就是计算机会根据一些计算，尽量保持多的信息下，选择最后降到几维，但是维度很大的话这个计算量很大的n_coments = [0,1]之间的浮点数，但是这时候必须指定另一个超参数：svd_solver='full' 。这个就是直接希望PCA保留多少的信息量，**这种方法比较好，直接自己指定保留多少信息**，不用画图探索，可以先上来保证80%的信息量，看看会保留几维特征，然后慢慢的加。

5.2 人脸数据集实战

好了，明白了sklearn中的PCA如何使用，下面我们来玩一个人脸识别数据集的降维，再来看一下降维的一些细节吧，这次用的数据集是sklearn的fetch_lfw_people：

```
# 导入包
from sklearn.datasets import fetch_lfw_people
from sklearn.decomposition import PCA

import matplotlib.pyplot as plt
import numpy as np


# 导入数据，并且探索一下子
faces = fetch_lfw_people(min_faces_per_person=60)
faces.images.shape   # (1348, 64, 47)  1348张图片，每张64*47
faces.data.shape    # (1348, 2914)  这是把上面的后两维进行了合并，共2914个特征（像素点）


# 下面我们先可视化一下子这些图片，看看长什么样
fig, axes = plt.subplots(3, 8, figsize=(8,4), subplot_kw={"xticks":[], "yticks":[]})

for i, ax in enumerate(axes.flat):
    ax.imshow(faces.images[i, :, :], cmap='gray')
```

下面我们使用PCA进行降维处理

```
pca = PCA(150).fit(faces.data)  # 降到150维
V = pca.components_   # 这就是那组基
```

```
V.shape          #（150,2914） 每一行是一个基，用这个乘上我们样本X，就会得到降维后的结果矩阵
```

上面的components_属性就是提取的150维之后的那组基，也就是数学公式里面的特征向量。我们可以可视化一下这个，亲身感受一下这组基是什么样子的（也就是PCA从原始数据上提取了怎么样的主成分特征）

```
# 下面可视化一下V
fig, axes = plt.subplots(3, 8, figsize=(8,4), subplot_kw={"xticks":[], "yticks":[]})
for i, ax in enumerate(axes.flat):
    ax.imshow(V[i,:].reshape(62, 47), cmap='gray')
```

结果如下：

这个怎么跟鬼一样？不要害怕， 其实这就是PCA提取的主成分特征，虽然可能看不出模样，但是你会发现，PCA提取特征的时候，会更加关注五官特征，眼睛轮廓，嘴，鼻子等还是比较明显的，这也挺符合我们现在的人脸识别原理，我们识别人脸，不正是主要看五官的区别吗？ 所以PCA在这方面提取的特征还算是合理的。

那么你可能会想了，PCA的这个降维过程能不能再回去啊？其实还真有个这样的接口函数pca.inverse_transform， 这个可以实现降维的逆转，但是真的完全逆转吗？ 我们可以做一个实验：

```
# 我们先得到降维后的数据
X_dr = pca.transform(faces.data)     # 这个是1358,150的数据

#然后我们调用接口逆转
```

```
X_inverse = pca.inverse_transform(X_dr)
X_inverse.shape    # （1348， 2914） 看这个形状还真回去了啊
```

我们看这个形状，还真是回去了啊，但是真回去了吗？

```
# 下面对比一下pca的逆转和原来图片的区别
fig, ax = plt.subplots(2, 10, figsize=(10,2.5), subplot_kw={"xticks":[], "yticks":[]})
for i in range(10):
    ax[0,i].imshow(faces.images[i,:, :], cmap='binary_r')
    ax[1,i].imshow(X_inverse[i].reshape(62, 47), cmap="binary_r")   # 降维不是完全可逆的
```

我们可以发现，逆转回去的图片是模糊的，虽然能够基本认出来。

> ★ Inverse_transform的功能：是基于X_dr的数据进行升维，将数据重新映射到原数据所在的空间中，而并非恢复所有的数据。但我们也看出，降维到150以后的数据，的确保留了原数据的大部分信息，所以图像看起来，才会和原数据高度相似，只是稍微模糊罢了。 ”

所以说**降维是不可能完全逆转的**， 抛弃掉的那些信息，很难找回来了，但是这样我们更好理解PCA的常用应用了，原来我们现实的人脸识别系统，什么汽车站，火车站，我们放上身份证，为啥能那么快就能判断出来是不是自己啊， 其实就是用了PCA的技术，因为用150个特征就可以判断是不是本人，完全不需要原来的2914个特征啊，这样计算量不就减少了？

6. 总结

首先，从生活的场景出发，感受了一下什么叫做降维，然后一个例子阐述PCA究竟在干什么事情，总结起来就是PCA找主成分的时候其实在寻找K个尽可能的把样本区分开的方向，并且这个K个方向互不相关，这样就可以做到在保留尽可能多的数据信息的情况下把数据的维度降到了K维。然后就是在这个基础上讲了一下PCA的数学原理，理解了一下PCA是怎么去找K个互不相关的方向的。最后用鸢尾花的例子实现了PCA的数学计算过程，然后又认识了sklearn中的PCA，然后用这个方式对人脸识别的数据集进行降维和逆转，并进行对比。

总之，学习完之后希望还是有些帮助吧，之前也说过PCA是一个很有用的技术手段，一般用在数据特征处理的部分，是特征工程的部分，那么最后再来说一个细节：和普通的特征选择有什么不同呢？**特征选择是从已存在的特征中选取携带信息最多的，选完之后的特征依然具有可解释性，而PCA，将已存在的特征压缩，降维完毕后不是原来特征的任何一个，也就是PCA降维之后的特征我们根本不知道什么含义了.**

参考：

- https://www.sohu.com/a/206848524_314987
- https://www.cnblogs.com/onemorepoint/p/8688484.html
- https://blog.csdn.net/wuzhongqiang/article/details/101117219
- https://blog.csdn.net/Marvelous_Morty/article/details/89303165