**浙江工商大学**

**本科生期末大作业报告**

**报告题目: 数据挖掘分类预测实验**

**课程名称： Python与数据科学实验**

**专业名称： 计科1902**

**学 号： 1935010102**

**姓 名： 徐超信**

**授课教师： 宋超**

**成 绩：**

**日 期 ： 2021 年6 月 20 日**

# 第1章 数据挖掘与分类预测

## 发展现状与趋势

 数据科学的研究:将数据学的理论和方法应用于许多领域，从而形成专门领域的数据学，例如：脑数据学、行为数据学、生物数据学、气象数据学、金融数据学、地理数据学等等。

数据挖掘技术:是一个充满希望的研究领域,越来越多人已经认识到数据挖掘技术能将原始数据转换为有意义的形式及其应用的潜在价值,每年都有新的数据挖掘方法和模型问世,人们对它的研究正日益广泛和深入。作为一个新兴的信息检索技术,数据挖掘存在着很多亟待解决的问题,如数据挖掘算法的有效性和可扩展性、数据的时序性和其他系统集成等。

随着数据挖掘技术不断被应用到各个领域和各种算法不断被应用到数据挖掘领域中,将更大激发数据挖掘技术的潜力,进一步推进数挖掘技术的发展和应用。

## 数据挖掘中的分类预测任务与目标

### 数据挖掘的任务

数据挖掘的任务有关联分析、聚类分析、分类分析、异常分析、特异群组分析和演变分析等

分类预测(Classification and Prediction)：

数据挖掘中的分类(classification)任务，是在若干样本数据上，学习到一

个模型，然后用这个模型对新数据进行预测（分类）。

分类的目的是获得一个分类函数或分类模型(也常常称作分类器)，该模

型能把数据样本映射到某一个给定类别

有: KNN、GNB、Logistic回归、决策树、SVM、线性回归、神经网络等算法

## 应用价值:

数据挖掘日益受到人们的关注，并已成为当前计算机领域的一大热点，其研

究重点也逐渐从发现方法转移到系统应用，并且注重多种发现策略和技术的集

成，以及多学科之间的相互渗透。

由于数据挖掘带来的显著的经济效益，最先是应用于金融和工商业领域。它

们都在利用数据挖掘技术帮助管理客户生命周期的各个阶段，包括争取新的客

户、在已有客户的身上赚更多的钱和保持好的客户。如果能够确定好的客户的特

点，那么就能为客户提供针对性的服务。比如，已经发现了购买某一商品的客户

的特征，那么就可以向那些具有这些特征但还没有购买此商品的客户推销这个商

品；找到流失的客户的特征就可以在那些具有相似特征的客户还未流失之前进行

针对性的弥补，因为保留一个客户要比争取一个客户成本低。保险公司和证券公

司也开始采用数据挖掘来减少欺诈。经销商更多的使用数据挖掘来决定每种商品

在不同地点的库存，通过数据挖掘更灵活的使用促销和优惠券手段。这些都是数

据挖掘的具体应用。现在人们不仅能够迅速接收

新生事物，更希望能用所掌握的信息去“预测不可预测的未来”，去更加准确地

把握自己的命运。对信息社会中的任何组织或个人来说，其最大的资本就是所掌

握的“有用”信息，要想不被信息的海洋所淹没，并在激烈的竞争中拔得头筹，

就必须武装自身信息处理能力。

# 第2章 分类模型与数据挖掘库sklearn

## KNN最近邻方法

sklearn.neighbors 提供了 neighbors-based (基于邻居的) 无监督学习以及监督学习方法的功能。 无监督的最近邻是许多其它学习方法的基础，尤其是 manifold learning (流行学习) 和 spectral clustering (谱聚类)。 neighbors-based (基于邻居的) 监督学习分为两种： classification （分类）针对的是具有离散标签的数据，regression （回归）针对的是具有连续标签的数据。

最近邻方法背后的原理是从训练样本中找到与新点在距离上最近的预定数量的几个点，然后从这些点中预测标签。 这些点的数量可以是用户自定义的常量（K-最近邻学习）， 也可以根据不同的点的局部密度（基于半径的最近邻学习）。距离通常可以通过任何度量来衡量： standard Euclidean distance（标准欧式距离）是最常见的选择。Neighbors-based（基于邻居的）方法被称为非泛化机器学习方法， 因为它们只是简单地”记住”了其所有的训练数据

尽管它简单，但最近邻算法已经成功地适用于很多的分类和回归问题:

例如手写数字或卫星图像的场景。 作为一个 non-parametric（非参数化）方法，它经常成功地应用于决策边界非常不规则的分类情景下。

sklearn.neighbors 可以处理 Numpy 数组或 <cite>scipy.sparse</cite> 矩阵作为其输入。 对于密集矩阵，大多数可能的距离度量都是支持的。对于稀疏矩阵，支持搜索任意的 Minkowski 度量。

**最近邻分类**

**最近邻分类属于 *基于实例的学习* 或 *非泛化学习* ：它不会去构造一个泛化的内部模型，而是简单地存储训练数据的实例。 分类是由每个点的最近邻的简单多数投票中计算得到的：一个查询点的数据类型是由它最近邻点中最具代表性的数据类型来决定的。**

**scikit-learn 实现了两种不同的最近邻分类器： 基于每个查询点的 个最近邻实现，**

**其中 [k](https://img.cntofu.com/book/scikit-learn-doc-zh/docs/img/f93871977da52a6d11045d57c3e18728.jpg) 是用户指定的整数值。**[**RadiusNeighborsClassifier**](https://www.cntofu.com/book/170/docs/generated/sklearn.neighbors.RadiusNeighborsClassifier.html#sklearn.neighbors.RadiusNeighborsClassifier)**基于每个查询点的固定半径 [r](https://img.cntofu.com/book/scikit-learn-doc-zh/docs/img/451ef7ed1a14a6cdc38324c8a5c7c683.jpg) 内的邻居数量实现， 其中 [r](https://img.cntofu.com/book/scikit-learn-doc-zh/docs/img/451ef7ed1a14a6cdc38324c8a5c7c683.jpg) 是用户指定的浮点数值。**

**-邻居分类是 下的两种技术中比较常用的一种。 值的最佳选择是高度依赖数据的：**

**通常较大的 [k](https://img.cntofu.com/book/scikit-learn-doc-zh/docs/img/f93871977da52a6d11045d57c3e18728.jpg) 是会抑制噪声的影响，但是使得分类界限不明显。**

**如果数据是不均匀采样的，那么**[**RadiusNeighborsClassifier**](https://www.cntofu.com/book/170/docs/generated/sklearn.neighbors.RadiusNeighborsClassifier.html#sklearn.neighbors.RadiusNeighborsClassifier)**中的基于半径的近邻分类可能是更好的选择。**

**用户指定一个固定半径 ，使得稀疏邻居中的点使用较少的最近邻来分类。**

**对于高维参数空间，这个方法会由于所谓的 “维度灾难” 而变得不那么有效。**

## 朴素贝叶斯

朴素贝叶斯方法是基于贝叶斯定理的一组有监督学习算法，即“简单”地假设每对特征之间相互独立。 给定一个类别 [y](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/0775c03fc710a24df297dedcec515aaf.jpg) 和一个从 [x_1](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/f15b9946d9078749f894a78579dc6778.jpg) 到 [x_n](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/14f6506599a88a5297ea712fa70eece4.jpg) 的相关的特征向量， 贝叶斯定理阐述了以下关系:

[P(y \mid x_1, \dots, x_n) = \frac{P(y) P(x_1, \dots x_n \mid y)}{P(x_1, \dots, x_n)}](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/32f500a4e2eba65727c1e003699dff90.jpg)

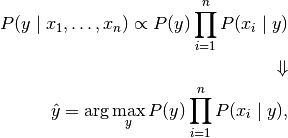
使用简单(naive)的假设-每对特征之间都相互独立:

[P(x_i | y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i | y) ,](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/9fd83615429a9be9e5698d35bec8642a.jpg)

对于所有的 :i 都成立，这个关系式可以简化为

[P(y \mid x_1, \dots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \dots, x_n)}](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/1c12ea7ea179efd16ce513645034d41a.jpg)

由于在给定的输入中 [P(x_1, \dots, x_n)](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/03dc262433e357325639af531c5bf70e.jpg) 是一个常量，我们使用下面的分类规则:

[](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/983133e80141fbf289a10f379c11b34f.jpg)

我们可以使用最大后验概率(Maximum A Posteriori, MAP) 来估计 [P(y)](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/d41288778c3d66bcae947c3078469126.jpg) 和 [P(x_i \mid y)](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/db23fadfab6b660dbfa2934c4536beb1.jpg) ; 前者是训练集中类别 [y](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/0775c03fc710a24df297dedcec515aaf.jpg) 的相对频率。

各种各样的的朴素贝叶斯分类器的差异大部分来自于处理 [P(x_i \mid y)](https://github.com/apachecn/sklearn-doc-zh/blob/master/docs/master/img/db23fadfab6b660dbfa2934c4536beb1.jpg) 分布时的所做的假设不同。

尽管其假设过于简单，在很多实际情况下，朴素贝叶斯工作得很好，特别是文档分类和垃圾邮件过滤。这些工作都要求 一个小的训练集来估计必需参数。(至于为什么朴素贝叶斯表现得好的理论原因和它适用于哪些类型的数据，请参见下面的参考。)

相比于其他更复杂的方法，朴素贝叶斯学习器和分类器非常快。 分类条件分布的解耦意味着可以独立单独地把每个特征视为一维分布来估计。这样反过来有助于缓解维度灾难带来的问题。

另一方面，尽管朴素贝叶斯被认为是一种相当不错的分类器，但却不是好的估计器(estimator)，所以不能太过于重视从 predict\_proba 输出的概率。

## 高斯朴素贝叶斯

[GaussianNB](https://www.cntofu.com/book/170/docs/generated/sklearn.naive_bayes.GaussianNB.html#sklearn.naive_bayes.GaussianNB) 实现了运用于分类的高斯朴素贝叶斯算法。特征的可能性(即概率)假设为高斯分布:

[P(x_i \mid y) &= \frac{1}{\sqrt{2\pi\sigma2_y}} \exp\left(-\frac{(x_i - \mu_y)2}{2\sigma^2_y}\right)](https://img.cntofu.com/book/scikit-learn-doc-zh/docs/img/92c470d884176abfcc29a5218ccf0aef.jpg)

## logistic 回归

logistic 回归，虽然名字里有 “回归” 二字，但实际上是解决分类问题的一类线性模型。在某些文献中，logistic 回归又被称作 logit 回归，maximum-entropy classification（MaxEnt，最大熵分类），或 log-linear classifier（对数线性分类器）。该模型利用函数 [logistic function](https://en.wikipedia.org/wiki/Logistic_function) 将单次试验（single trial）的可能结果输出为概率。

scikit-learn 中 logistic 回归在 [LogisticRegression](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#sklearn.linear_model.LogisticRegression) 类中实现了二分类（binary）、一对多分类（one-vs-rest）及多项式 logistic 回归，并带有可选的 L1 和 L2 正则化。

注意，scikit-learn的逻辑回归在默认情况下使用L2正则化，这样的方式在机器学习领域是常见的，在统计分析领域是不常见的。正则化的另一优势是提升数值稳定性。scikit-learn通过将C设置为很大的值实现无正则化。

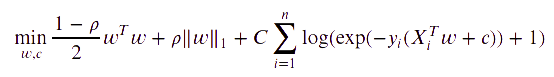
作为优化问题，带 L2罚项的二分类 logistic 回归要最小化以下代价函数（cost function）：

[\underset{w, c}{min,} \frac{1}{2}w^T w + C \sum_{i=1}^n \log(\exp(- y_i (X_i^T w + c)) + 1) .](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/203c5a2c58d6567a86dbc86faa92209e.jpg)

类似地，带 L1 正则的 logistic 回归解决的是如下优化问题：

[\underset{w, c}{min,} |w|1 + C \sum{i=1}^n \log(\exp(- y_i (X_i^T w + c)) + 1) .](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/d7ff3091308658ce388554d420581459.jpg)

Elastic-Net正则化是L1 和 L2的组合，来使如下代价函数最小:

[](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/021new1.jpg)

其中ρ控制正则化L1与正则化L2的强度(对应于l1\_ratio参数)。

## 梯度提升决策树

## Gradient Tree Boosting（梯度树提升）

[Gradient Tree Boosting](https://en.wikipedia.org/wiki/Gradient_boosting) 或梯度提升回归树（GBRT）是对于任意的可微损失函数的提升算法的泛化。 GBRT 是一个准确高效的现有程序， 它既能用于分类问题也可以用于回归问题。梯度树提升模型被应用到各种领域，包括网页搜索排名和生态领域。

GBRT 的优点:

* 对混合型数据的自然处理（异构特征）
* 强大的预测能力
* 在输出空间中对异常点的鲁棒性(通过具有鲁棒性的损失函数实现)

GBRT 的缺点:

* 可扩展性差（校对者注：此处的可扩展性特指在更大规模的数据集/复杂度更高的模型上使用的能力，而非我们通常说的功能的扩展性；GBRT 支持自定义的损失函数，从这个角度看它的扩展性还是很强的！）。由于提升算法的有序性(也就是说下一步的结果依赖于上一步)，因此很难做并行.

模块 [sklearn.ensemble](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/classes.html#module-sklearn.ensemble) 通过梯度提升树提供了分类和回归的方法.

## 随机森林

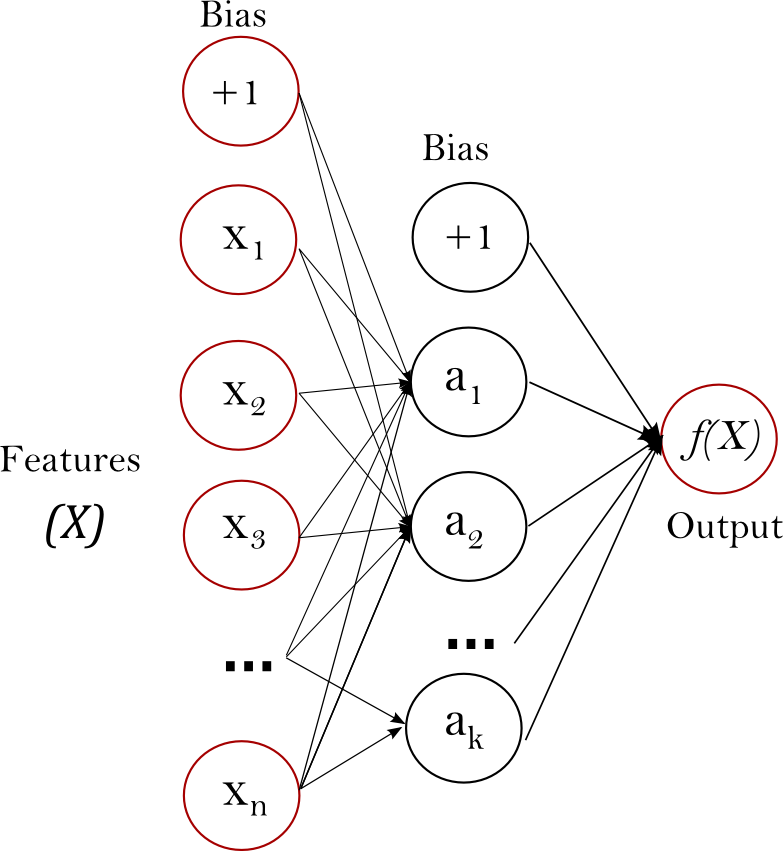
在随机森林中集成模型中的每棵树构建时的样本都是由训练集经过有放回抽样得来的（例如，自助采样法-bootstrap sample，这里采用西瓜书中的译法）。

另外，在构建树的过程中进行结点分割时，选择的分割点是所有特征的最佳分割点，或特征的大小为 max\_features 的随机子集的最佳分割点。

这两种随机性的目的是降低估计器的方差。的确，单棵决策树通常具有高方差，容易过拟合。随机森林构建过程的随机性能够产生具有不同预测错误的决策树。通过取这些决策树的平均，能够消除部分错误。随机森林虽然能够通过组合不同的树降低方差，但是有时会略微增加偏差。在实际问题中，方差的降低通常更加显著，所以随机森林能够取得更好地效果。

## 多层感知器

**多层感知器（MLP）** 是一种监督学习算法，通过在数据集上训练来学习函数 [f(\cdot): R^m \rightarrow R^o](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/fd6f65ce4fb7491d7628d1ce576c19d4.jpg)，其中 [m](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/94156b879a7455cb0d516efa9c9c0991.jpg) 是输入的维数，[o](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/f7f0b321634c8d80ceacdc75ee3c68b6.jpg) 是输出的维数。 给定一组特征 [X = {x_1, x_2, ..., x_m}](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/d7228aff11bb03497e40badd984560a6.jpg) 和标签 [y](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/0775c03fc710a24df297dedcec515aaf.jpg) ，它可以学习用于分类或回归的非线性函数。 与逻辑回归不同的是，在输入层和输出层之间，可以有一个或多个非线性层，称为隐藏层。 图1 展示了一个具有标量输出的单隐藏层 MLP。

[](http://sklearn.apachecn.org/cn/0.19.0/_images/multilayerperceptron_network.png)

**图1：单隐藏层MLP.**

最左层的输入层由一组代表输入特征的神经元 [{x_i | x_1, x_2, ..., x_m}](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/3cc550ecff73666ed35ae1efee48b4f4.jpg) 组成。 每个隐藏层中的神经元将前一层的值进行加权线性求和转换 [w_1x_1 + w_2x_2 + ... + w_mx_m](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/f1fb5834480bfa9770be94da12bbd514.jpg) ，再通过非线性激活函数 [g(\cdot):R \rightarrow R](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/80a5660d27392922e501744cab3623da.jpg) - 比如双曲正切函数 tanh 。 输出层接收到的值是最后一个隐藏层的输出经过变换而来的。

该模块包含公共属性 coefs\_ 和 intercepts\_ 。 coefs\_ 是一系列权重矩阵，其中下标为 [i](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/43e13b580daefe5ba754b790dfbd216c.jpg) 的权重矩阵表示第 [i](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/43e13b580daefe5ba754b790dfbd216c.jpg) 层和第 [i+1](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/4aafe42b7f9cf8d06d93b9246d01bbfd.jpg) 层之间的权重。 intercepts\_ 是一系列偏置向量，其中的下标为 [i](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/43e13b580daefe5ba754b790dfbd216c.jpg) 的向量表示添加到第 [i+1](https://github.com/apachecn/sklearn-doc-zh/blob/53e5a60fc5b15e7b81c00f17e4983f9de25d7f26/docs/master/img/4aafe42b7f9cf8d06d93b9246d01bbfd.jpg) 层的偏置值。

多层感知器的优点：

* 可以学习得到非线性模型。
* 使用partial\_fit 可以学习得到实时模型（在线学习）。

多层感知器的缺点：

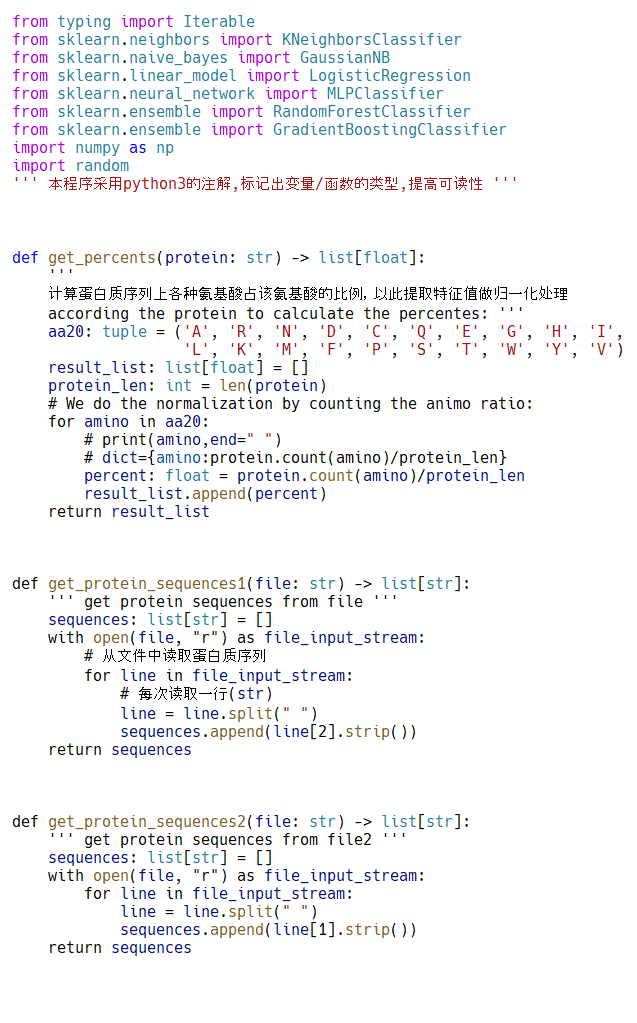
* 具有隐藏层的 MLP 具有非凸的损失函数，它有不止一个的局部最小值。 因此不同的随机初始化权重会导致不同的验证集准确率。
* MLP 需要调试一些超参数，例如隐藏层神经元的数量、层数和迭代轮数。
* MLP 对特征归一化很敏感。

多层感知器对特征的缩放是敏感的，所以它强烈建议您归一化您的数据。

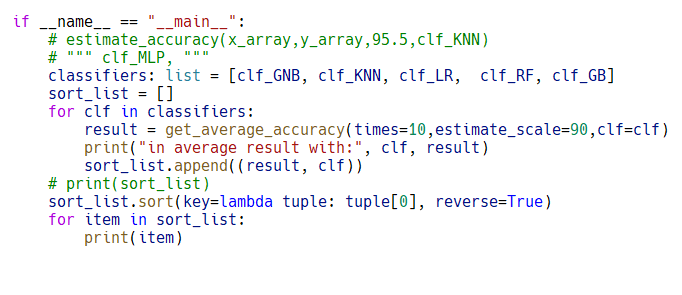
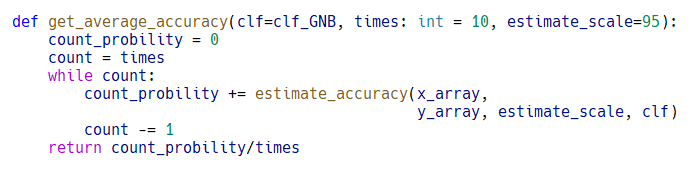
例如，将输入向量 X 的每个属性放缩到到 [0, 1] 或 [-1，+1] ，或者将其标准化使它具有 0 均值和方差 1。注意，为了得到有意义的结果，您必须对测试集也应用相同的尺度缩放

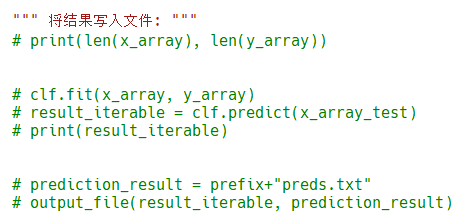
# 第3章 蛋白质功能分类实验

## Python code









## The result:

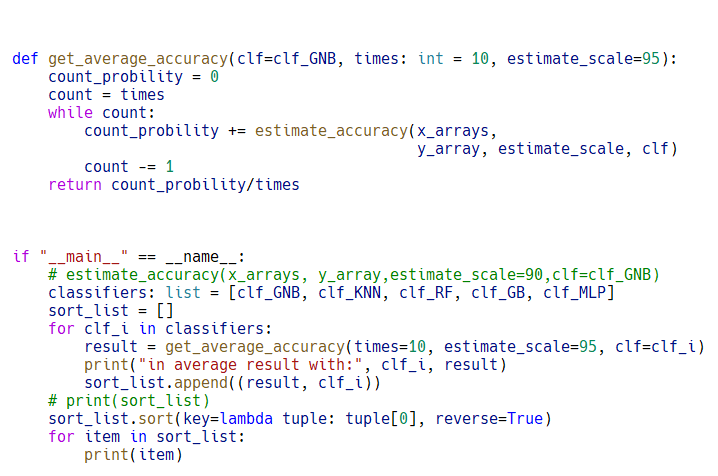
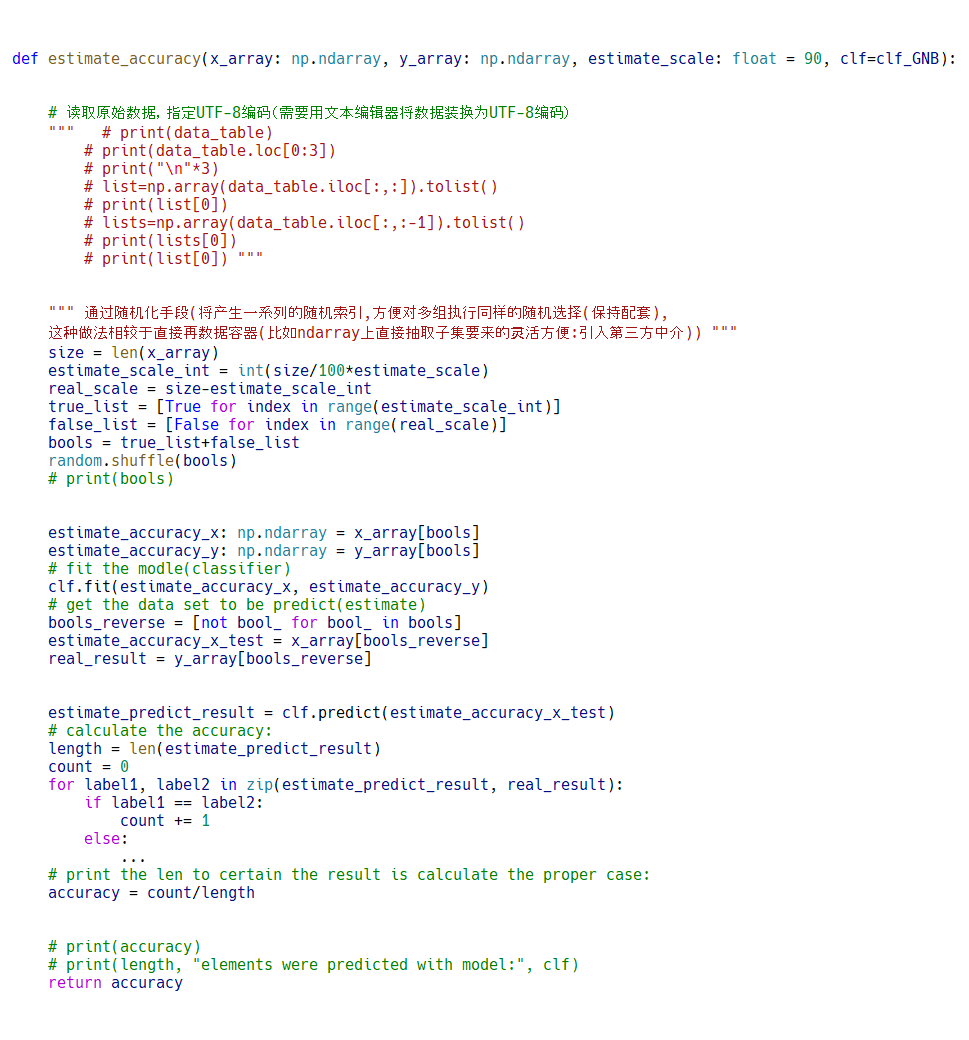
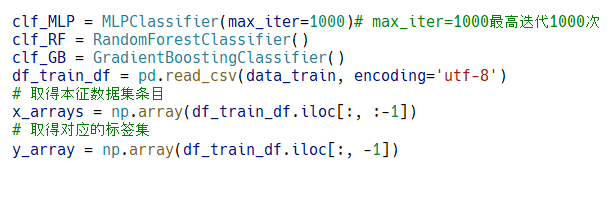


可见,在该预测问题中,表现较好的有和随机森林RandomForestClassifier分类器和逻辑回归分类器LogisticRegressionClassifier分类器

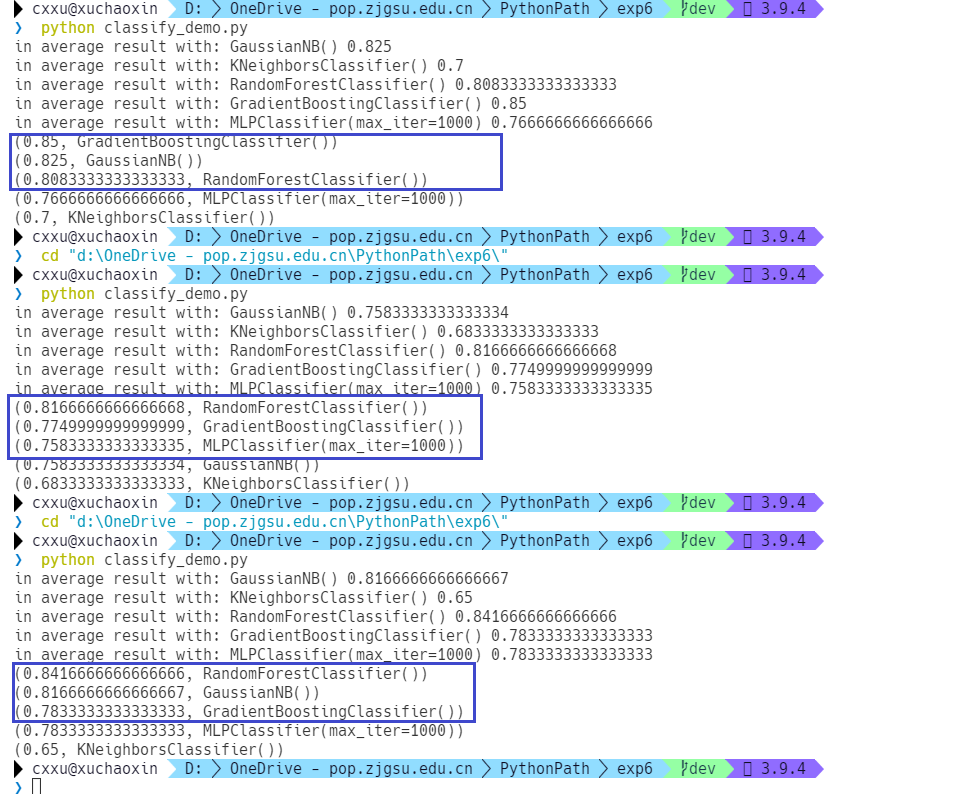
# 第4章 重大疾病预测

## 初始版本:采用独立随机化比较

### The code



### The result:



运行参数说明:

每中分类器预测采用随机化方式预测十次(用95%的数据预测余下5%的数据)

该方案中没有采用严格的控制变量,函数def get\_average\_accuracy

存在较大耦合，将bool数值的生成单独提取出来，封装成generate\_random\_bools()以供灵活调用需求于是有了第二版：

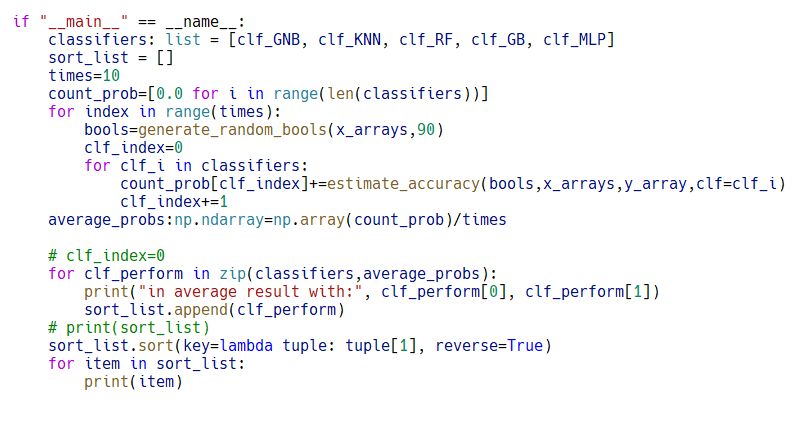
## 第二版本:

采用控制变量+随机化取样来估计各分类器在该问题的分类表现:

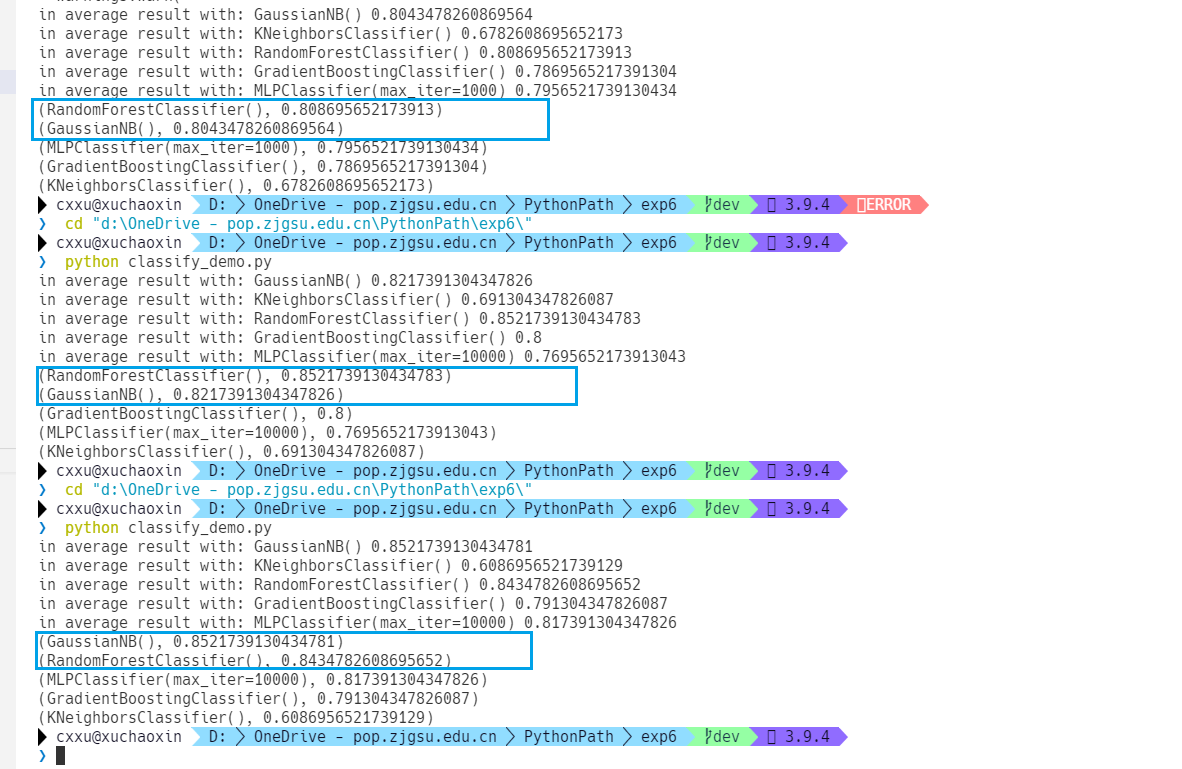
### The code







### The result



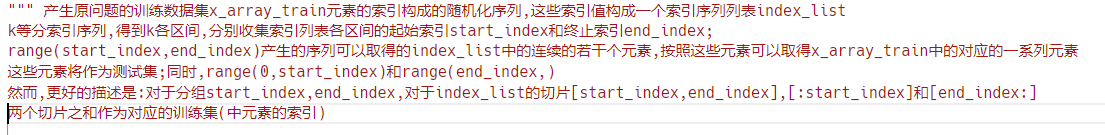
### The conclusion(结论)

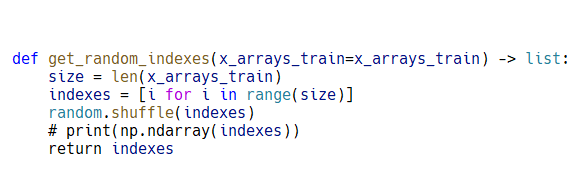
在该分类问题中，随机森林的预测效果和高斯朴素贝叶斯的预测效果相对较好

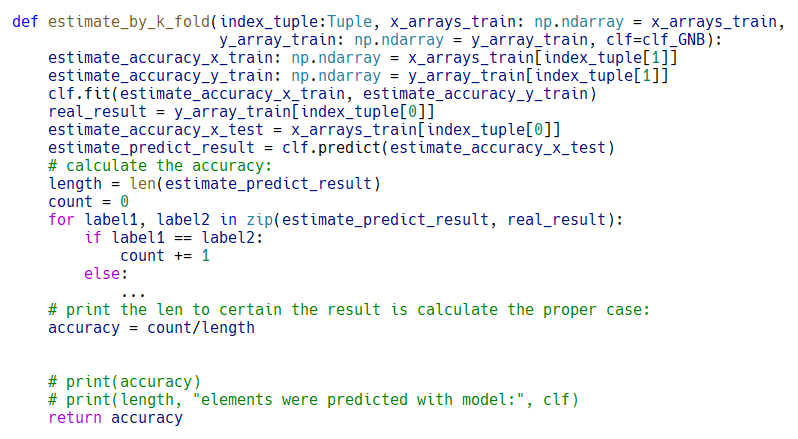
在控制输入变量一致的前提下考察不同分类器的在该问题的分类表现，评价结果过较为稳定

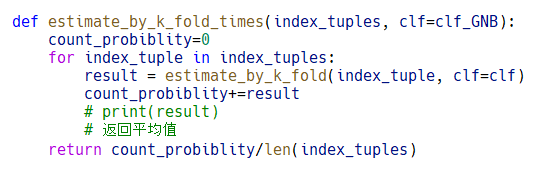
## 基于k折叠交叉验证法:

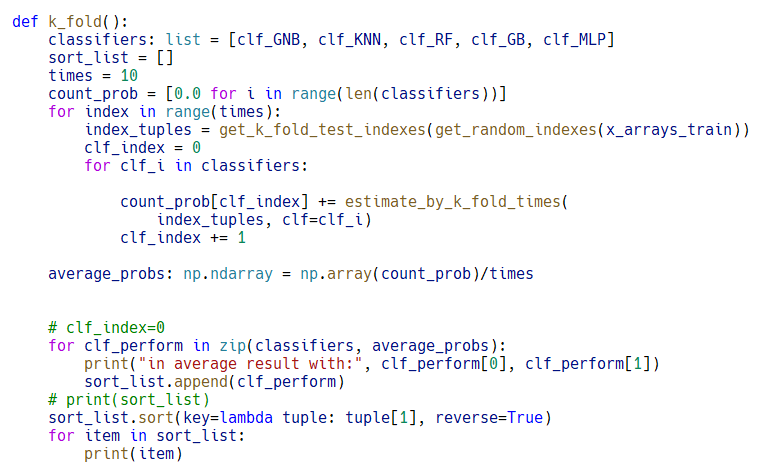
输入一致的前提下对比分类器的分类效果





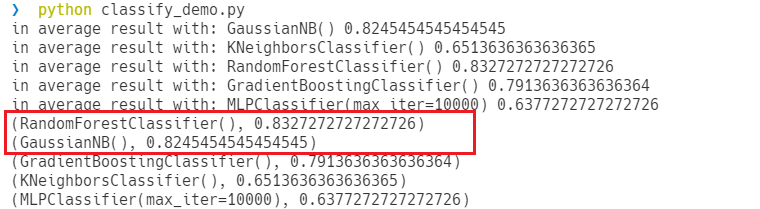




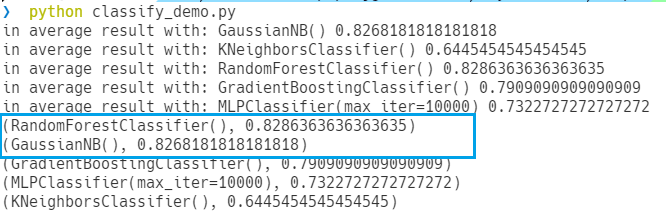


### The result:

K=10运行结果(计算量大)



K=5:运行结果



结论,通过K折叠交叉验证法的计算结果可知,随机森林和高斯朴素贝叶斯方法的预测效果较好

# 第5章 实验工作及其创新点总结、数据挖掘经验总结、未来学习计划与数据科学实践展望。

## 经验总结：

通过本学期的分类问题的处理，我在数据挖掘方面有了初步，发现了数据挖掘的作用和价值，以及在挖掘价值的过程中带来的乐趣，让我切实的感受到学有所用：许多之前认为不实用的数学知识和实验思想在这几次数据挖掘实验中都一定程度的排上了用场，说明基础理论知识还是不可以轻视的，特别是数据挖掘和数据分析工作，需要严谨地设计计算方案，公证合理的效果分析，尽量避免偶然性

此外还有自己所编写代码也需要具有良好的可读性，python语言方便易用，凭借着一点，管受欢迎，但同时，我们不难发现，过度弱化类型，不利于代码的可读性，在代码到达一定规模的时候，我们那面会怀念静态语言具有的清晰的类型；幸运的是，python3为我们提供了注解（包括函数注解，变量注解，以及泛型）尽管这些特性不带有强制性，但是大大提高了python代码的可读性和编写过程中的准确性，IDE也能够为我们提供更多更详细的信息，减少错误的函数调用

另一方面就是关于分类器模型的效果评价（模型预测准确性的估算）

模型效果的估算策略多种多样：

* 留出法(hold out): 直接将数据集D划分为两个互斥的集合，其中一个集合作为训练集S,另一个作为测试集T。
* 交叉验证法(cross validation): 也称为K折交叉验证(K-fold cross validation)

1. 准备好训练集，随机分成K等份，即K个子集；一般情况，K=5或10.
2. 将其中1个子集作为测试集，其余K-1个子集合并作为训练集。

每个子集均作为测试集一次，如此交叉验证重复K次，

平均K次结果最终得到一个单一估测。

* 留一法(leave-one-out): 特殊的交叉验证法，即K=N(样本总量)
* 自助法(boostrap)

给定N个样本的数据集D, 每次从D中随机挑选一个样本，将其放入D’,然后再放回数据集D，使得样本在下次采样时仍有可能被采到,这个过程重复执行N次后，就会得到包含N个数据样本的集合D’。

我主要采用随机化的留出法，同时使用交叉验证法来进一步证实分析结果的可靠性.通过一个随机化bool列表，配合numpy巧妙实现随机取样/分组取样

对于交叉验证法，可以创建一个索引号数组，在用random.shuffle()来随机化，同时通过索引(切片)来截取测试集;

单一分类器中，采用多次随机采样求平均值（减少误差），在不同的分类器间采用输入控制变量法（公平性）

代码从最初的高耦合，经过通过多次解耦，已经可以较为灵活的改变模型评价方案进行新的测评

同时，我们发现，不同的分类器在不同的分类问题上的分类效果的相对优劣性可能会发生变化，没有在每种分类问题下都能够保持最好的效果的分类器

## 学习计划

数据挖掘涵盖的内容不少，为了更好更熟练的利用个工具，我认为有必要在今后查阅相关专业资料增强自身在数据挖掘方面的能力，提升工作效率

## 展望：

数据挖掘是如此有用而且有趣，具有现在看来还是相当有发展前景和应用前景，相信将来一定会受益于数据挖掘工作所带来的好处