上海对外经贸大学

2021 -2022 第一学期

《应用多元统计分析》

案 例 报 告

报告名称：	基于个人信息的收入范围分析
学　　院：	统计与信息学院
专　　业：	数据科学与大数据技术
学　　号：	18076014
学生姓名：	陈郁欣
课程教师：	李文
课程编号：	376.007.201

2021　年　12 月

基于个人信息的收入范围分析

摘要

个人收入一直是国民经济体中重要的一部分，政府可以根据收入范围变化进行政策的实施。本文主要使用了美国 1994 年人口普查数据库中的个人信息以及工资范围数据，使用朴素贝叶斯分类器和线性判别分类器，对个人收入范围进行了分类，通过实例阐述收入分类模型以及细分模型的实现过程，验证了模型训练有效性，双模型的训练结果为 0.80 和 0.84，均大于零精度，模型效果较好，为相关机构根据信息预测个人收入提供了理论基础与建议。

关键词： 收入分类 朴素贝叶斯模型 线性判别模型

一、理论基础

个人收入是指一个国家一年内个人得到的全部收入，个人从各种途径所获得的收入的总和，反映了该国个人的实际购买力水平，同时也预示了未来消费者对于商品、服务等需求的变化。个人收入指标是预测个人的消费能力，未来消费者的购买动向及评估经济情况的好坏的一个有效指标。

一个社会的个人收入的上升或下降可以观察出经济情况是否有好转或者衰退的征兆，这种对收入的判断对货币的汇率走势的影响也是非常重大的。如果个人收入上升速度过快，可能会出现通货膨胀的问题，央行就会考虑加息，会对货币汇率产生很强势的效应。

通常决定一个人收入水平的因素有很多，例如学历情况、工作的职业、工作的地理位置、人种、国籍、是否有犯罪记录、是否结婚、性别等，这些不同的因素以某种方式联系在一起，继而成为了判断个人收入范围的一系列指标，对这些个人信息的多方面分析，可以初步对收入进行判断。同时，由于科技的快速发展，人工智能、5G等各种新兴技术出现，社会进入了一个大数据时代，对信息的获取的途径增多，信息量也大大增大，为了使得信息与科技利用率最大化，就要使用合理的模型进行数据有效分析。

通过对收入的合理判断，政府可以有效控制货币汇率走势，避免经济方面出现误判；同时也可以使用收入类别范围分析，来决定对个人实行什么类型的福利、是否允许贷款、划分缴费梯度等，便于合理制定正确的决策，确保社会正常运行。

二、研究内容

本文以“基于个人信息的收入范围分析”为题，以部分不同国家的成人年收入范围为研究对象，通过多维度的个人信息数据，结合朴素贝叶斯分类器模型、线性判别分类器模型来对个人收入进行量化分析，对个人所有已知信息进行相关数据挖掘并分类收入范围，为政府决策提供一定的理论支持，同时也便于企业、银行等相关机构细分群体，通过对不同的个人实施不同类型的服务或执行不同类型的限制，提升办事效率，提高效益，降低风险。

本文的研究主要涵盖以下三个方面：

（1） 首先完成对个人收入范围分类模型的确认，找到适合的模型并构建。

（2） 对个人信息特征进行系统化的数据预处理工作，基于部分训练数据建立拟合模型并测试拟合效果。

（3） 将对剩余测试数据进行收入范围分类并进行结果汇总与正确率检查。

基于以上分析，可以针对模型得出的不同背景下的个人的不同收入范围，完成个人收入分类，从而达到便于制定政策、策略等目的。

三、模型原理

3.1 朴素贝叶斯分类模型

朴素贝叶斯分类器是一系列以假设特征之间强独立下运用的简单概率分类器，以贝叶斯定理为基础。朴素贝叶斯分类器的假设为，尽管样本特征相互依赖或者有些特征是由其他特征所决定的，它也认为这些样本特征在判断给定类别的概率分布上是独立的。其思想基础为：对于给定的训练集数据，在给出待分类项的情况下，首先基于特征条件独立假设学习输入输出的联合概率分布，然后基于此模型，对给定的输入，利用贝叶斯定理求解在此项出现的条件下各个类别出现的后验概率，哪个概率最大，就认为这个待分类项目属于哪个类别，这也被称为最大后验估计Maximum A Posteriori（MAP）。

设输入空间$\chi \subseteq \Re^n$是n维向量的集合，输出空间为类标记集合$\mathcal{Y} = \{c_1, c_2, \dots, c_{\mathrm{K}}\}$。输入为特征向量$x \in \chi$,输出为分类标记$y \in \mathcal{Y}$。$X$是定义在输入空间$\chi$上的随机向量，$Y$是定义在输出空间$\mathcal{Y}$上的随机变量。$P(X,Y)$是$X$和$Y$的联合概率分布。训练数据集$T$由$P(X,Y)$独立同分布产生，公式如下

$$T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$$

朴素贝叶斯模型通过对训练数据集进行计算，学习联合概率分布$P(X,Y)$。首先学习先验概率分布

$$P(Y = c_k), k = 1,2, \dots, K$$

与条件概率分布

$$P(X = x|Y = c_k) = P\left(X^{(1)} = x^{(1)}, \dots, X^{(n)} = x^{(n)} \middle| Y = c_k\right), k = 1,2, \dots, K$$

由此即可得到

$$P(x,y)=P(X=x|Y=c_k)P(Y=c_k)$$

其中，由于条件概率分布$P(X=x|Y=c_k)$有指数级数量的参数，估计实际不可行，故朴素贝叶斯法对条件概率分布作了条件独立性假设，假设用于分类的特征在类别确定的条件下都是条件独立的，假设公式如下

$$P(X=x|Y=c_k)=P\left(X^{(1)}=x^{(1)},\dots,X^{(n)}=x^{(n)}\middle|Y=c_k\right)$$

$$=\prod_{j=1}^{n}P\left(X^{(j)}=x^{(j)}\middle|Y=c_k\right)$$

朴素贝叶斯模型分类时，对给定的输入x，通过学习到的模型计算后验概率分布$P(Y=c_k|X=x)$，将计算出的后验概率最大值的那个类作为x的类输出，计算公式如下

$$P(Y=c_k|X=x)=\frac{P(X=x|Y=c_k)P(Y=c_k)}{\sum_k\left(P(X=x|Y=c_k)P(Y=c_k)\right)}$$

将模型的条件独立性假设代入上式可得

$$P(Y=c_k|X=x)=\frac{P(Y=c_k)\prod_j P\left(X^{(j)}=x^{(j)}\middle|Y=c_k\right)}{\sum_k\left(P(Y=c_k)\prod_j P\left(X^{(j)}=x^{(j)}\middle|Y=c_k\right)\right)}$$

由于需要对最大后验概率进行判断继而选择出合适的类别，故朴素贝叶斯分类模型可以表示为

$$y=f(x)=\arg max_{c_k}\frac{P(Y=c_k)\prod_j P\left(X^{(j)}=x^{(j)}\middle|Y=c_k\right)}{\sum_k\left(P(Y=c_k)\prod_j P\left(X^{(j)}=x^{(j)}\middle|Y=c_k\right)\right)}$$

在一个固定的数据集中，上式分母对所有类别c_k都是相同的，所以朴素贝叶斯分类器也可以用如下简化后的式子进行表示

$$y=f(x)=\arg max_{c_k}P(Y=c_k)\prod_j P\left(X^{(j)}=x^{(j)}\middle|Y=c_k\right)$$

由上述模型公式可得，针对给定训练数据，计算先验概率及条件概率后，对于给定的实例计算后验概率，最大化后验概率即可确定实例的值，继而对样本进行分类。

3.2 线性判别分类模型

线性判别分类器（Linear Discriminant Analysis）是对费希尔线性鉴别方法（FLD）的归纳，基本思想是将高维的模式样本投影到最佳鉴别矢量空间，以此达到抽取分类信息和压缩特征空间维数的效果，投影后保证模式样本在新的子空间有最大的类间距离和最小的类内距离，也就是空间中的样本在模式的划分下有最佳的可分离性。

线性判别分类模型以协方差、协方差矩阵以及散度矩阵为理论基础。

协方差分为随机变量的协方差和样本协方差。

随机变量协方差$cov(X,Y)$与方差相似，是分布的一个总体参数，是对两个随机变量联合分布线性相关程度的一种度量标准。两个随机变量线性相关性越高，协方差越大，完全线性无关则协方差为零。

$$cov(X,Y) = E[(X - E(X))(Y - E(Y))]$$

由于真实数据中，样本不同变量特性的数值尺度不相同，所以不能直接使用若干协方差的大小作为相关性强弱的比较，因为引入相关系数η对协方差进行归一化，相关系数的取值范围为$[-1,1]$。

$$\eta = \frac{cov(X,Y)}{\sqrt{var(X)var(Y)}}$$

样本协方差的定义则是，对于现有的N个样本，每个样本均具有n维属性，每一维属性都可以看作是一个随机变量，设每一个样本为$x_j = \left[x_j^{(1)}, x_j^{(2)}, \dots, x_j^{(n)}\right]$，计算两个属性之间的线性关系$q_{ab}$。

$$q_{ab} = \frac{\sum_{j=1}^{N}(x_j^{(a)} - \overline{x^{(a)}})(x_j^{(b)} - \overline{x^{(b)}})}{N-1}$$

这里分母为$N-1$是因为随机变量的数学期望未知，用样本均值代替，自由度减1。

由于考虑到变量不仅仅是两变量的，而可能是多维随机的，所以两变量协方差可以扩展为多维随机变量中任意两变量之间的协方差。

$$\Sigma = E[(X - E(X))(X - E(X))^T]$$

$$\hat{\Sigma} = \begin{bmatrix} q_{11} & \cdots & q_{1n} \\ \vdots & \ddots & \vdots \\ q_{n1} & \cdots & q_{nn} \end{bmatrix} = \frac{1}{N-1}\sum_{j=1}^{N}(\boldsymbol{x}_{\cdot j} - \overline{\boldsymbol{x}})(\boldsymbol{x}_{\cdot j} - \overline{\boldsymbol{x}})^T$$

其中公式中$\overline{\boldsymbol{x}}$为列向量样本均值，$\boldsymbol{x}_{\cdot j}$为样本集合中的第$j$个样本，同样为列向量。上文提及到变量特性数值尺度的问题，需要将不同变量之间的线性关系变换为可以互相比较的情形，所以对样本进行归一化后，最终样本协方差矩阵$\hat{\boldsymbol{\Sigma}}$的表达方式为

$$\hat{\Sigma} = \frac{1}{N-1}\sum_{j=1}^{N}\boldsymbol{z}_{\cdot j}\boldsymbol{z}_{\cdot j}^{T}$$

$$\boldsymbol{z}_{\cdot j} = \frac{\boldsymbol{y}_{i\cdot}}{\sigma_i}, \quad \boldsymbol{y}_{i\cdot} = \boldsymbol{x}_{\cdot j} - \overline{\boldsymbol{x}}$$

$\boldsymbol{y}_{i\cdot}$对样本进行平移，σ_i为维度i的标准差，消除了数值大小的影响。

散度矩阵公式为

$$S_i = \sum_{j=1}^{N} \boldsymbol{z}_{\cdot j} \boldsymbol{z}_{\cdot j}^{T}$$

类内散度矩阵$S_W = S_i + S_j$，类间散度矩阵$S_B = (\overrightarrow{\mu_i} - \overrightarrow{\mu_j})(\overrightarrow{\mu_i} - \overrightarrow{\mu_j})^T$

线性判别分类模型算法的本质为使用拉格朗日乘子法$max_{\omega} \frac{\omega^T S_B \omega}{\omega^T S_W \omega}$，将类内散度矩阵和类间散度投影到直线上，并得出到原点的距离。选择合适的向量使得比值最大。因此，LDA模型是一种有效的特征抽取的方法，使用这种方法进行数据处理后，可以使投影后模式样本的类间散度矩阵最大，类内散度矩阵最小，对数据进行有效分类。

四、模型求解

4.1 数据准备

本文使用的是美国1994年人口普查数据库中的成人个人信息以及工资范围数据（https://archive.ics.uci.edu/ml/machine-learning-databases/adult/），数据包含32561个个体的14种个人信息和工资范围，其中工资范围分成小于等于50K和大于50K两种类别，个人信息属性变量包含年龄，工种，学历，职业等重要信息，其中包含7个类别型变量，属性变量内容如下：

表1 个人信息类别属性

序号	属性名	含义
0	age	年龄
1	workclass	工作类型
2	fnlwgt	最终权重
3	education	教育程度
4	education_num	受教育时间
5	maritial_status	婚姻状况
6	occupation	职业
7	relationship	关系
8	race	种族
9	sex	性别
10	capital_gain	资本收益
11	capital_loss	资本损失
12	hours_per_week	每周工作小时数

部分数据示例如下：

表2 数据集部分数据示例

age	workclass	fnlwgt	education	marital_status	occupation	capital_loss	hours_per_week	native_country	income
39	State-gov	77516	Bachelors	Never-married	Adm-clerical	0	40	United-States	<=50K
50	Self-emp-not-inc	83311	Bachelors	Married-civ-spouse	Exec-managerial	0	13	United-States	<=50K
38	Private	215646	HS-grad	Divorced	Handlers-cleaners	0	40	United-States	<=50K

4.1.1 数据预处理

为了不影响后续分类效果，首先需要对数据进行空缺值检查，检查结果为无空缺值。但是在观察类别型变量时，发现部分变量里包含“？”这一类，此类情况也可以被视作空缺值，可以使用出现频率最高的类别填充或者直接删除此行，考虑到使用其他类别填充仍然会在一定情况上影响模型判断结果，并且异常情况较少，故采用直接删除的预处理方法。

前文提及到，个人信息属性中包含了七种类别型变量，字符串在模型中是无法直接进行计算的，必须通过变量编码转成相应的数值型向量才可以继续进行模型求解。由于数据中的类别型变量不存在次序关系，使用简单的整数编码会给类别添加自然的顺序关系，从而导致结果不佳或者得到以外的结果，所以这里本文使用One-hot编码进行处理，其方法是使用N位状态寄存器来对N个状态进行编码，每个状态都有独立的寄位器位，并且在任意时候只有其中一位是有效的。这个向量的表示为一项属性的特征向量，也就是同一时间只有一个激活点不为0，其他都是0。转换后的数据集变量数从13扩充到了104列，部分转换后情况如下：

表3 类别型变量One-hot转换示例

workclass_1	workclass_2	workclass_3	workclass_4	workclass_5	workclass_6	workclass_7
1	0	0	0	0	0	0
1	0	0	0	0	0	0
0	1	0	0	0	0	0
1	0	0	0	0	0	0
1	0	0	0	0	0	0

通过对所有数据的观察，可以看出，部分数据是千万级数据，但是还有一部

分范围只在0~1之间，差别过大，直接进行模型计算的话，绝对值很大的属性可能对类别起到很大的决定性作用，在学习算法中占主导位置，会造成一定偏差，影响最终分类结果，所以最后需要对数据进行标准化处理，提升模型精度，这里我们使用中位数$median$和四分位矩IQR转换，使用此方法的好处是在缩放的同时还可以剔除异常值，处理公式为：

$$v_i' = \frac{v_i - median}{IQR}$$

4.2 朴素贝叶斯分类模型求解

4.2.1 模型拟合

这里本文使用Python的机器学习库sklearn中的GaussianNB函数，划分训练集进行模型拟合，并使用测试集进行分类。

4.2.2 效果评估

测试集部分分类结果与真实结果对比如下：

表4 朴素贝叶斯分类部分结果对比

pred	true
<=50K	<=50K
<=50K	<=50K
>50K	<=50K
>50K	>50K
<=50K	<=50K

模型对测试集的分类准确率为0.8005，为了检验模型效果，需要对零精度进行计算，零精度是指通过预测最频繁的类可以达到的精度，即把类别全归为出现次数最多的那个工资范围可以达到的准确率，计算得出测试集中工资大于等于50K的人数为7953，零精度为0.7533，故可以得出结论，模型在分类类别标签的方面完成较好。

同时可以使用混淆矩阵来总结分类算法性能，在评估分类模型性能时可能出现四种类型的结果，TP（预测正类，实际正类）、TN（预测负类，实际负类）、FP（预测正类，实际负类）、FN（预测负类，实际正类），绘制模型混淆矩阵可视化图

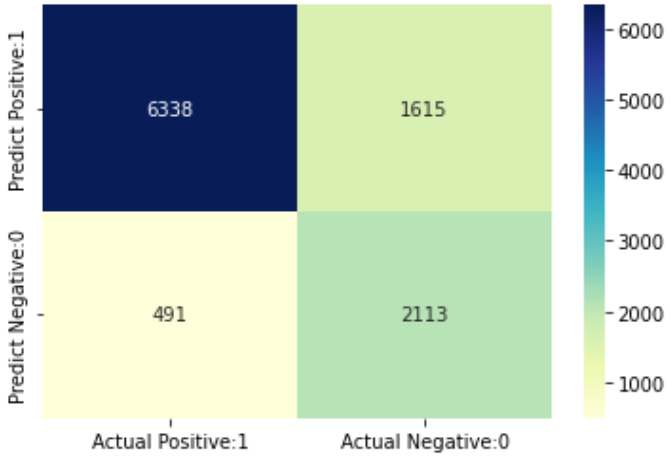

图1 朴素贝叶斯分类结果混淆矩阵

通过四项分类结果，可以计算正确预测正结果的比例精度Precision，在所有实际的正类中正确预测正类的比例召回率Recall以及精度和召回率的加权调和平均值F1-score，各指标计算结果如下

表5 朴素贝叶斯分类模型效果

	precision	recall	f1-score
<=50K	0.93	0.80	0.86
>50K	0.57	0.81	0.67
准确率	0.80		

由指标可以看出，总体分类效果较佳，在分类大于50K这一类的时候可能出现偏差较大。此时使用另一个直观衡量分类模型性能的工具ROC曲线，绘制出不同阈值水平的真阳性率和假阳性率。ROC曲线下面积ROC-AUC为0.8891，再次印证了模型效果较佳。

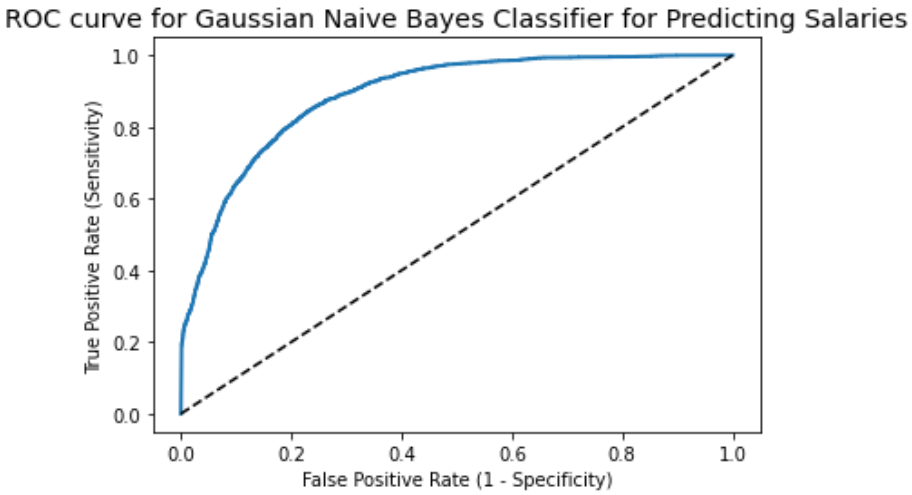

图2 朴素贝叶斯分类结果ROC曲线图

4.3 线性判别分类模型求解

4.3.1 模型拟合

这里本文使用Python的机器学习库sklearn中的LinearDiscriminantAnalysis函数，划分训练集进行模型拟合，并使用测试集进行分类。

4.3.2 效果评估

测试集部分分类结果与真实结果对比如下：

表6 线性判别分类部分结果对比

pred	true
<=50K	<=50K
<=50K	<=50K
<=50K	<=50K
>50K	>50K
<=50K	<=50K

模型对测试集的分类准确率为0.8381，同样远大于零精度0.7533，模型在分类类别标签的方面完成较好。绘制LDA模型预测混淆矩阵可视化图，并且计算各项指标，绘制ROC曲线图，ROC曲线下面积ROC-AUC为0.8869。

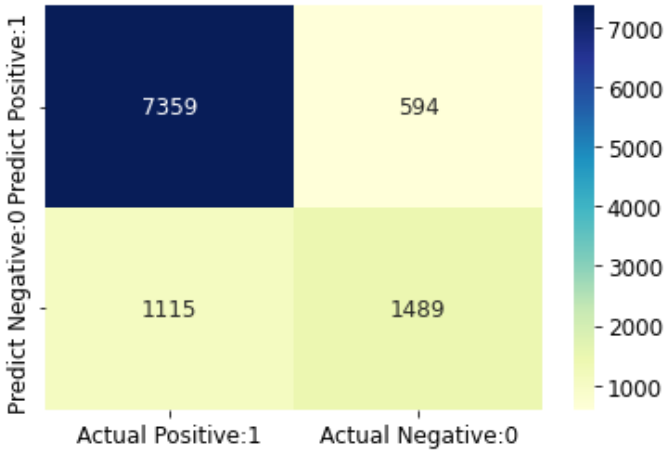

图3 线性判别分类结果混淆矩阵

表7 线性判别分类模型效果

	precision	recall	f1-score
<=50K	0.87	0.93	0.90
>50K	0.71	0.57	0.64
准确率	0.84		

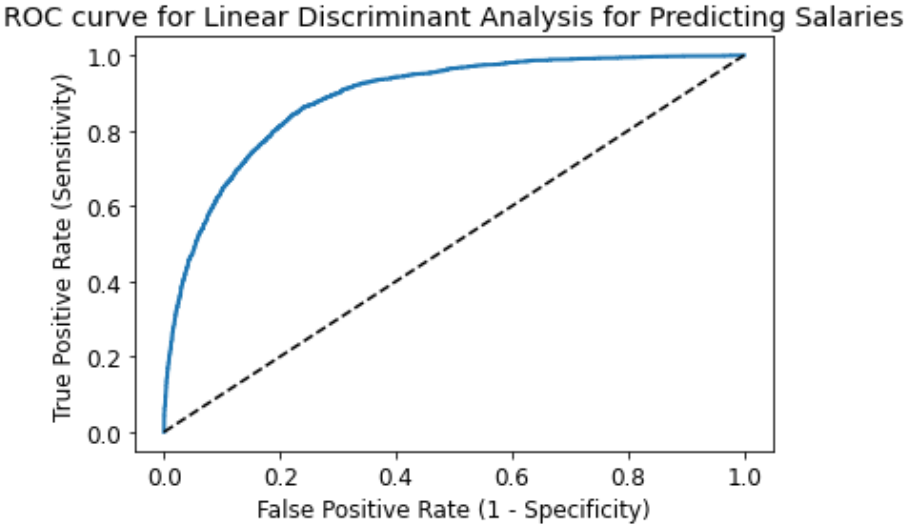

图4 线性判别分类结果ROC曲线图

由可视化以及图表数据可证明，模型效果较佳。

五、结论

本文建立了朴素贝叶斯分类器和线性判别分类器模型来预测一个人的年收入是否超过50K，两个模型都具有良好的性能，模型精度分别为0.80和0.84，精度大于零精度0.75，所以不存在直接归为频率高的类别的现象，并且两个模型在测试集上的精度与在训练集上相似，没有过度拟合的迹象，因此可以得出结论，模型在预测类标签方面做的很好。同时，两个模型的ROC-AUC皆接近于1，在预测一个人年收入是否超过50K方面做的很好。但是，在对两个类别标签的精度和召回率分别进行计算出，可以观察到超过50K标签的预测效果较差于不超过50K标签，可能因为数据集大部分的样本收入都不超过50K，导致数据分布不均匀，影响模型学习结果。

通过两个分类器，可以较好地针对一些个人信息对收入进行分类预测，以此达到便于进行决策的目的，同时，模型和数据也都存在部分问题，如朴素贝叶斯假设不符合实际情况、数据分布不均匀等，可以在后续的分析中对数据内容进行调整，并且可以结合其他模型进行更精确的预测分类，根据收入范围制定合理决策，确保社会正常运行。

六、参考文献

[1] LDA（线性判别分类器）学习笔记 - Luke_Ye - 博客园. (2019). Retrieved 5 December 2021, from https://www.cnblogs.com/LukeStepByStep/p/10529706.html

[2] 协方差与协方差矩阵 - 苦力笨笨 - 博客园. (2016). Retrieved 5 December 2021, from https://www.cnblogs.com/terencezhou/p/6235974.html

[3] kaggle-2 美国人口普查年收入 50K 分类_常思考->有目标->重实践->善反思-CSDN 博客_fnlwgt. (2017). Retrieved 5 December 2021, from https://blog.csdn.net/haluoluo211/article/details/78943332

[4] One-Hot 编码_邱邱邱的博客-CSDN 博客. (2019). Retrieved 5 December 2021, from https://blog.csdn.net/Artoria_QZH/article/details/103254171?spm=1001.2101.3001.6661.1&utm_medium=distribute.pc_relevant_t0.none-task-blog-2~default~CTRLIST~default-1.opensearchhbase&depth_1-utm_source=distribute.pc_relevant_t0.none-task-blog-2~default~CTRLIST~default-1.opensearchhbase

[5] BANERJEE, P. (2020). Naive Bayes Classifier in Python. Retrieved 5 December 2021, from https://www.kaggle.com/prashant111/naive-bayes-classifier-in-python/notebook

[6] 李航. (2019). 统计学习方法. 北京: 清华大学出版社.

七、附录

```
#导入包
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt # for data visualization purposes
import seaborn as sns # for statistical data visualization
%matplotlib inline

#导入数据
data = '/content/drive/MyDrive/adult.csv'
df = pd.read_csv(data, header=None, sep=',\s')
#加入列名
col_names = ['age', 'workclass', 'fnlwgt', 'education', 'education_num',
'marital_status', 'occupation', 'relationship',
            'race', 'sex', 'capital_gain', 'capital_loss', 'hours_per_week',
'native_country', 'income']
df.columns = col_names
df.head(3)
#查看是否有空缺值
df.info()
df.shape

#数据处理
#类别变量
#把文本型数据分出来
categorial = []
for var in df.columns:
  if df[var].dtype=='object':
    categorial.append(var)
print(categorial)
#观察每列不同类别的一个频率分布情况
for var in categorial:
  print(df[var].value_counts()/np.float(len(df)))
#从数据频率分布可以看出**workclass**, **occupation** 和 **native_country**都包
```

含一个缺失的类别，所以需要把含有这些值的行删掉，否则影响后续分类

```
df['workclass'].replace('?',np.NaN,inplace=True)
df['occupation'].replace('?',np.NaN,inplace=True)
df['native_country'].replace('?',np.NaN,inplace=True)
df.isnull().sum()
df.dropna(inplace=True)#不返回东西，只修改 data 里的值
df.isnull().sum()

#数值型变量
numerical = []
for var in df.columns:
  if df[var].dtype!='object':
    numerical.append(var)
print(numerical)
df[numerical].head()
df[numerical].isnull().sum()

#分特征向量和目标变量
X=df.drop(['income'],axis=1)
y=df['income']

#划分训练集和测试集
from sklearn.model_selection import train_test_split
X_train,             X_test,             y_train,             y_test             =
train_test_split(X,y,test_size=0.35,random_state=511)
X_train.shape,X_test.shape

#变量编码
import category_encoders as ce
encoder = ce.OneHotEncoder(cols=cate)
X_train = encoder.fit_transform(X_train)
X_test = encoder.transform(X_test)
X_train.head()

#特征缩放
```

```
cols = X_train.columns
from sklearn.preprocessing import RobustScaler
scaler = RobustScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
#转回数据框
X_train=pd.DataFrame(X_train,columns=[cols])
X_test=pd.DataFrame(X_test,columns=[cols])

#建立模型
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(X=X_train,y=y_train)

from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
lda=LDA()
lda.fit(X_train,y_train)

#预测
y_pred_NB = gnb.predict(X_test)
y_pred_NB
y_pred_lda = lda.predict(X_test)
y_pred_lda
Nb_result=pd.DataFrame()
Nb_result['pred']=list(y_pred_NB)
Nb_result['true']=list(y_test)
Nb_result.head(5)

lda_result=pd.DataFrame()
lda_result['pred']=list(y_pred_lda)
lda_result['true']=list(y_test)
lda_result.head(5)

#检查准确率
from sklearn.metrics import accuracy_score
```

```
print('Model accuracy score: {0:0.4f}'. format(accuracy_score(y_test, y_pred_NB)))
print('Model accuracy score: {0:0.4f}'. format(accuracy_score(y_test, y_pred_lda)))
y_pred_train_NB = gnb.predict(X_train)
y_pred_train_lda = lda.predict(X_train)
print('NB Training-set accuracy score: {0:0.4f}'. format(accuracy_score(y_train, y_pred_train_NB)))
print('LDA Training-set accuracy score: {0:0.4f}'. format(accuracy_score(y_train, y_pred_train_lda)))

#零精度是指通过预测最频繁的类可以达到的精度。
对比零精度以看我们模型的准确率是否好
y_test.value_counts()
null_accuracy = (7953/(7953+2604))
print('Null accuracy score: {0:0.4f}'. format(null_accuracy))

#混淆矩阵
from sklearn.metrics import confusion_matrix
cm1 = confusion_matrix(y_test, y_pred_NB)
cm_matrix = pd.DataFrame(data=cm1, columns=['Actual Positive:1', 'Actual Negative:0'],
                                 index=['Predict Positive:1', 'Predict Negative:0'])
sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
cm2 = confusion_matrix(y_test, y_pred_lda)
cm_matrix = pd.DataFrame(data=cm2, columns=['Actual Positive:1', 'Actual Negative:0'],
                                 index=['Predict Positive:1', 'Predict Negative:0'])
sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')

#各项指标
#NB
from sklearn.metrics import classification_report
```

```
print(classification_report(y_test, y_pred_NB))
#LDA
print(classification_report(y_test, y_pred_lda))

#ROC
# plot ROC Curve
from sklearn.metrics import roc_curve
y_pred1_NB = gnb.predict_proba(X_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, y_pred1_NB, pos_label = '>50K')
plt.figure(figsize=(6,4))
plt.plot(fpr, tpr, linewidth=2)
plt.plot([0,1], [0,1], 'k--' )
plt.rcParams['font.size'] = 12
plt.title('ROC curve for Gaussian Naive Bayes Classifier for Predicting Salaries')
plt.xlabel('False Positive Rate (1 - Specificity)')
plt.ylabel('True Positive Rate (Sensitivity)')
plt.show()

#ROC-AUC
#ROC-AUC 代表受试者工作特征-曲线下面积。它是一种比较分类器性能的技术。在这种技术中，我们测量曲线下的面积(AUC)。一个完美的分类器将有一个 ROC  AUC 等于 1，而一个纯粹的随机分类器将有一个 ROC AUC 等于 0.5。
#ROC 曲线下面积是 ROC 曲线下面积的百分比。
# compute ROC AUC
from sklearn.metrics import roc_auc_score
ROC_AUC = roc_auc_score(y_test, y_pred1_NB)
print('ROC AUC : {:.4f}'.format(ROC_AUC))

y_pred1_lda = lda.predict_proba(X_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, y_pred1_lda, pos_label = '>50K')
plt.figure(figsize=(6,4))
plt.plot(fpr, tpr, linewidth=2)
plt.plot([0,1], [0,1], 'k--' )
plt.rcParams['font.size'] = 12
plt.title('ROC curve for Linear Discriminant Analysis for Predicting Salaries')
```

```
plt.xlabel('False Positive Rate (1 - Specificity)')
plt.ylabel('True Positive Rate (Sensitivity)')
plt.show()

ROC_AUC = roc_auc_score(y_test, y_pred1_lda)
print('ROC AUC : {:.4f}'.format(ROC_AUC))
```