# 信用卡诈骗交易的检测

金融数据挖掘 | 2017.12.05

**摘要**

本文主要探究比较了在机器学习中不平衡数据集的处理方法。

处理不平衡数据的学习问题一般分为三类：①重采样构造平衡集②特征选取③改进分类算法。本文主要就①②展开做了比较。

文中以 Kaggle 的信用卡交易公开数据集为基础，对比了重采样方法中的 Tomek links 欠采样、SMOTE 过采样、以及混合采样（Tomek links + SMOTE）,以及基于正态分布、自动编码器、神经网络的特征选取方法之间的差异。

经过比较后发现，在数据严重不平衡时（本例 0.0017:1），重采样方法处理能力有限，而基于神经网络的特征提取法表现较好。

# 目录

# 1. 相关背景

**研究目的**

本篇报告意在通过尝试不同的方法，去解决一个实际问题：在大量银行信用卡用户的正常流水交易中，如何检测出其中隐含的极少数的诈骗交易。值得注意的是，本例不同于常规的二分类问题，考虑到诈骗交易的危害性，本例的目的不是以提高判别分类的准确率为目标，而是尽可能的提高对诈骗交易的识别率。进而，如何以最小的代价从大量的交易中识别出诈骗交易即为本文要解决的问题。

银行如何辨别出诈骗交易？

以商户为例做简单描述[1]：其基本审计过程为先筛选---再核查。

通过用户的一些数据如：刷卡扣率、注册资本、经营成本、征信信息等，以及用户的刷卡总额、单笔金额、信用卡使用率等等，来筛选出可疑用户，再追踪其交易，核查判别是否为非法交易（恶意套现等）。

传统方法在于人工成本高，效率低。

相应的，在大量数据基础上，出现一些机器学习的算法来试着完成诈骗交易的鉴别。例如前面同学讲过的]Logistics 回归，判别分析等多元统计方法，或者决策树、随机森林、支持向量机 SVM、朴素贝叶斯、KNN 等等。并比较了各种算法的准确率、回收率等等。

**数据来源**

数据来源 Kaggle：https://www.kaggle.com/dalpozz/creditcardfraud

本文数据集共记录了某银行两天内的 284807 条流水交易记录，其中被标记为诈骗交易（非法交易）的共计 492 例，即在总数约 28.5W 的数据集中，只有约 500 例负类数据，数据集严重不平衡，负类样本量只有总的样本量的 0.0017。

数据说明：数据的维度为 284807*31, 31 个变量中，除了交易时间（实际记录的两天类每两笔交易之间的时间间隔）、金额和标签（normal/fraud）外，其他的 28 个特征变量均为从所有原始银行用户信息中提取的主成分。

利用主成分处理原始数据有两个好处：

1、很好的保护了银行用户的个人信息。这是一项很好的脱敏技术，在将敏感数据交于第三方处理是，这个方法就很好的避免了用户数据泄露的问题；
2、简化了数据集的结构。对于繁琐的用户信息和交易信息，提取主成分之后可以很大程度的简化数据集的结构，使得分析研究过程更加的高效。

不同于前面同学的案例，本例的特点和问题更接近于真实情况：数据量较大，数据类既不平衡、变量信息不完整等等。

# 2. 方法概述

## 2.1 关于分类器的评价体系

评价分类器性能的指标一般为分类器的准确率（accuracy）

定义为：对于给定的数据集，分类器的正确分类的数于样本总数之比。

而对于二分类问题，我们常用的评价指标是精确率（precision）与召回率（recall）,通常把关注的类归为正类，其他类为负类，分类器在测试集上的预测或者分类是否正确分为以下四种情况：

TP-----将正类预测为正类

FN-----将正类预测为负类

FP-----将负类预测为正类

TN----将负类预测为负类

精确率（precision） $P=\frac{TP}{TP+FP}$ ，

召回率（recall） $R=\frac{TP}{TP+FN}$ ，

此外，还有一个 F1 值，定义为精确率和召回率的调和平均：

$F_1=\frac{2TP}{2TP+FP+FN}$ ，

在 P 和 R 都相对较高的时候，F1 的值也相应的会高。

在本例中，我们的目标是在尽可能少地将正常交易误判为诈骗交易的情况下，提高将诈骗交易识别为诈骗交易的准确率。

由于我们的目标是从大量正常流水中识别出诈骗交易，故我们除了正常的计算正类（正常交易）的精准率和回收率之外，我们也要考虑负类（诈骗交易）的回收率和精准率。换句话说，诈骗交易（负类）的识别对我们的收益会更大。即我们的评价指标除了合法交易的精准率和回收率外，还要计算诈骗交易的精准率和回收率。即：

$$P(fraud)=\frac{TN}{TN+FN}, R(fraud)=\frac{TN}{TN+FP}.$$

## 2.2 问题及方法

问题主要集中在两个方面：

1. 数据集严重不平衡（0.0017:1）；
2. 需要同时保证正类和负类的 Recall 及 Precision。

关于不平衡数据集的学习问题有过很多研究， 大体上，对于不平衡数据集的机器学习问题有以下几种解决方法[2]：

1. 重新采样

对现有的不平衡数据集重新采样，来构造新的较为平衡的数据集。包括对多数类的欠采样、对少数类的过采样、以及混合采样等。

2. 特征选择

样本数量分布不平衡时，一般其特征分布也会不平衡。如何根据不平衡分类问题的特点，找到具有区分能力的特征量，来提高高稀有类的分别率。

3. 算法层面

通过改进算法和模型，使用不同的机器学习算法来提高分类的 Recall 等指标；或者是不改变算法架构，而去调整正类和负类错分的惩罚项，例如给予关键类的错分更高的惩罚值。

对于分类问题前面有很多同学讲过，介绍过很多种方法，也做过很多比较。这篇文章的主要工作就是在特征选择和重新采样上。

# 3. 算例分析

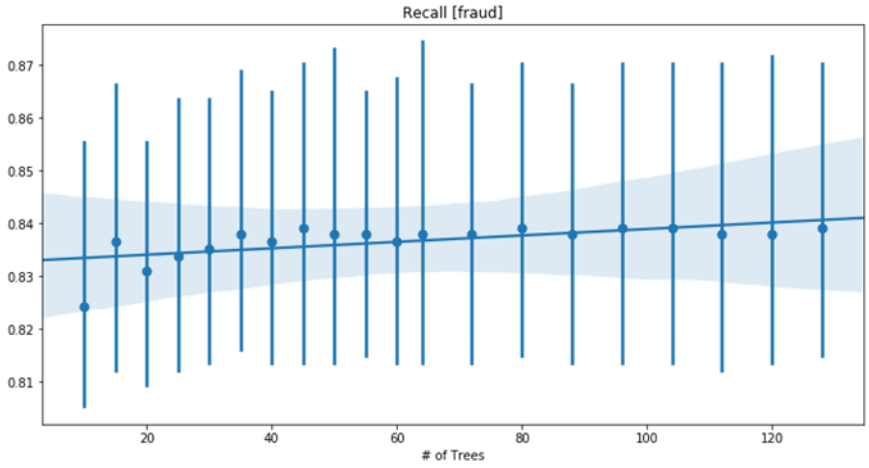
## 3.1. 利用重采样技术来处理非平衡数据的学习

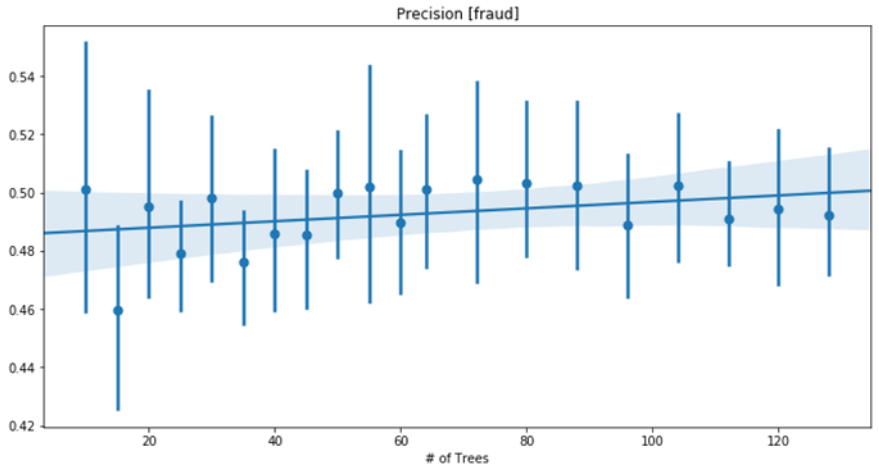
### 3.1.1 参数设定

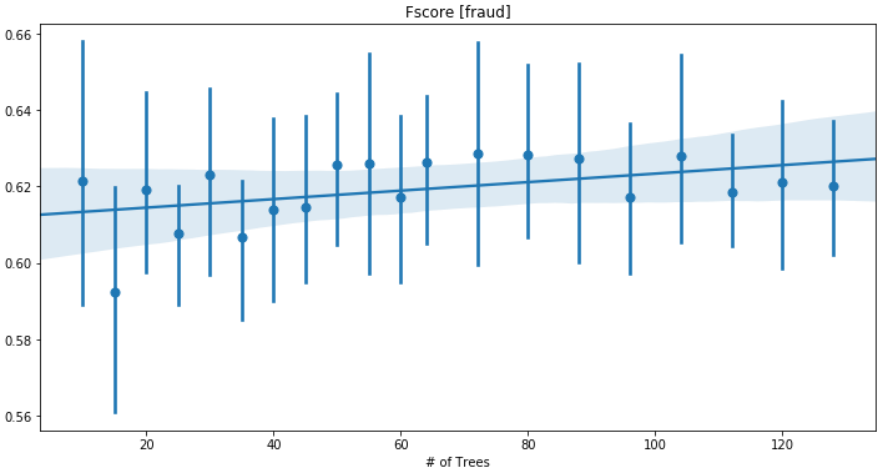
为了更好地比较各种重采样的方法之间的区别，我们统一了分类方法为随机森林算法，下面通过两个简单的实验，来探究下随机森林参数的设定。主要确定随机森林中决策树的数目和训练集中多数类和少数类样本的比例。

**决策树的数目**

主要是随机森林决策树的数目对最后分类的 Recall 等参数的影响。我们在 0-128 的范围内进行了简单的遍历，重复了五次，其结果对比如下：





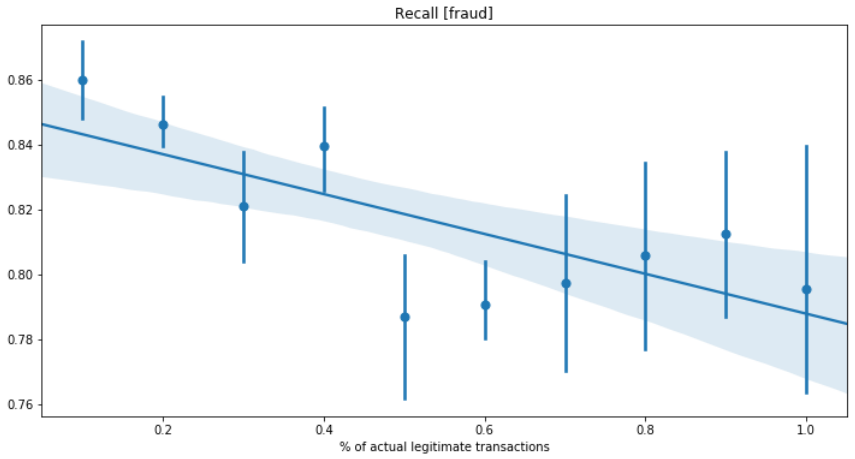



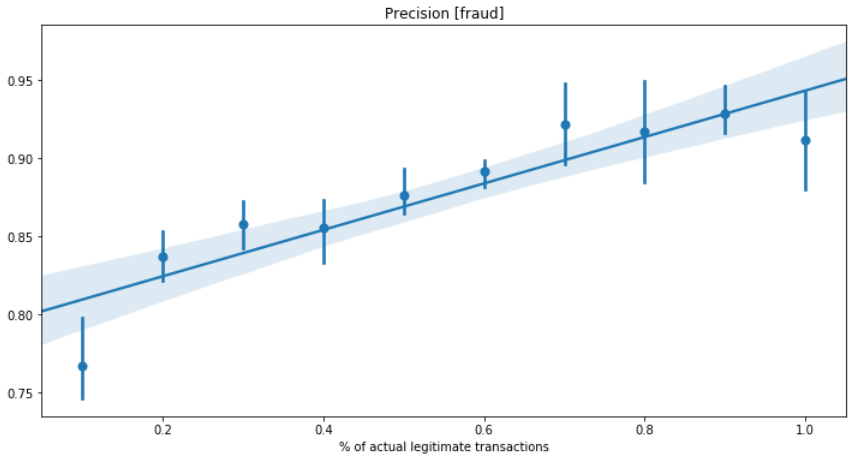

可以看到，随着决策树的棵树的增多，其精度和回收率都有所增加，但整个提升幅度并不大，考虑到，本例的样本量不是特别大，我们忽略随着树的棵树增多带来的计算的成本的提高，选择 N=128，尽可能的提高算法的精确度。

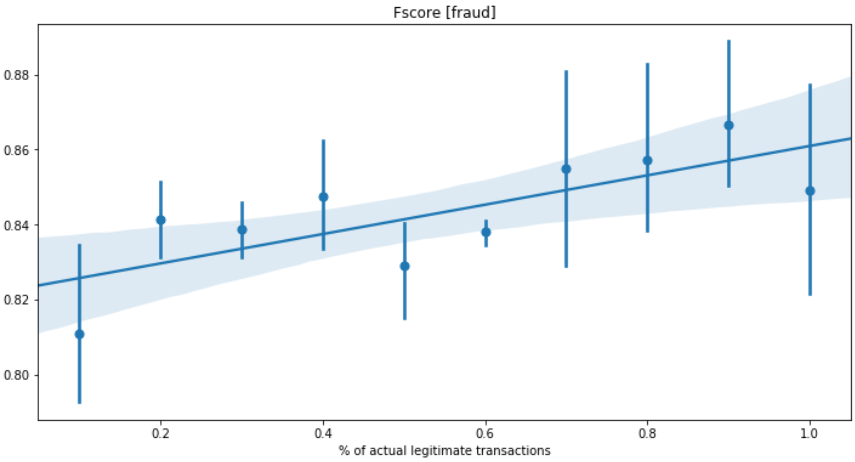
**训练集样本的比例**

因为多数类样本的数量较多，我们用简单的随机欠采样方法就是从多数类样本中随机的选取一定数量（或比例）的样本，再和少数类样本点组成新的固定比例的数据集，然后用 N=128 的随机森林去测试不同样本类比例的区别，同样重复五次，减少随机误差。

下图中横坐标显示的是多数类样本在新的训练集中所占比例的多少，从 0 到 1，间隔为 0.1，采用的分类算法为随机森林（决策树 N=128）。



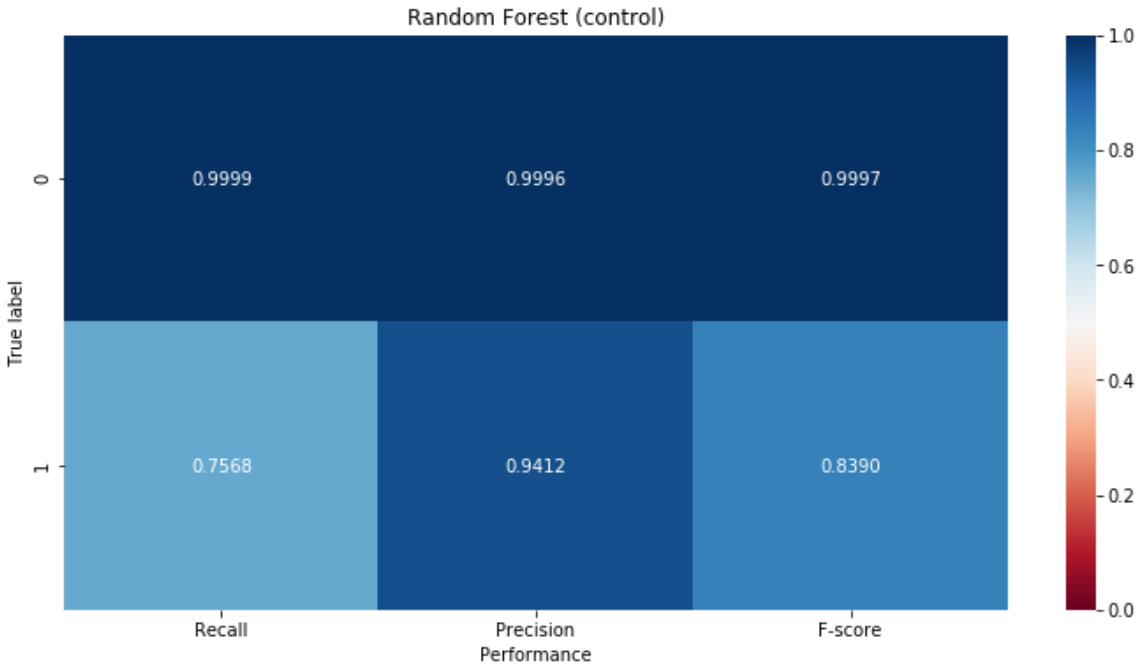




与我们想象的不一样的是，在二分类的各个参考指标上，并不是正类和负类的比例为 1:1 的时候，各项指标参数最高。而且，随着正类的样本比例的增多，负类的 Recall 是负相关的，而 Precision 和 F-score 是正相关的。

**对照组**

我们取了一组 N=128 的随机森林，不采取任何重采样的方法，去做一遍分类，并把该结果作为对照组（control）。




Running time: 79.99623427160623 seconds.

Accuracy: 0.999497

Recall: 0.756757

Precision: 0.941176

Fscore: 0.838951

### 3.1.2 TOMEK LINKS 欠采样方法

**定义：**假设样本点 $x_i$ 和 $x_j$ 属于不同的类别， $d(x_i, x_j)$ 表示两类别之间的距离。

如果不存在第三个样本点 $x_l$ 使得 $d(x_i, x_l) < d(x_i, x_j)$ 或者 $d(x_l, x_j) < d(x_i, x_j)$ 成立，

就称 $(x_i, x_j)$ 组成一个 Tomek link 对。

直观上去理解，如果两个样本点组成一个 Tomek link 对，要么其中的一个样本点为噪声点，要么两个样本点都处在边界上。寻找 Tomek links 对的方法就是 KNN，找到最邻近的不同类别的样本点。

在欠采样的处理方法中，找到 Tomek links 对，然后将其中的多数类的样本点给删除掉，这样就有效的减少了有干扰的多数类样本点或者是噪声点，从而达到欠采样的目的。

#另一方面，利用 Tomek links 对，我们也可以将整个 Tomek link 对的两个数据都删除掉，从而可以减少边界点或者噪声点待分类器性能带来的干扰。

Tomek links 欠采样算法流程：

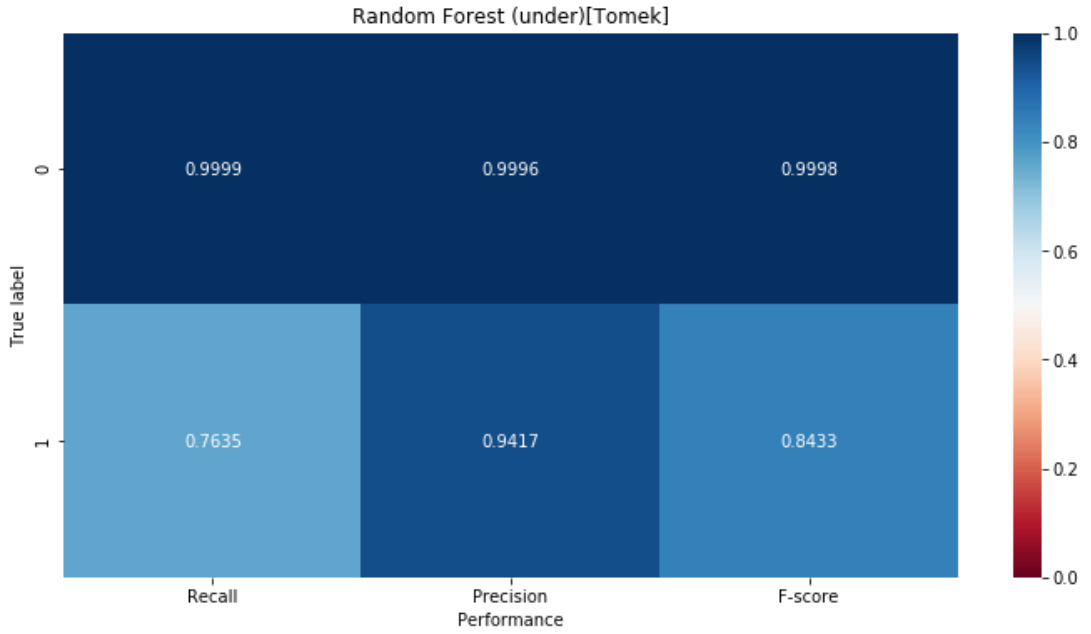
1、初始化训练集 T（T 应该包括所有的少数类样本和一定比例的多数类样本）；

2、用训练集 T 训练一个 1-NN 分类器（KNN 的近邻数为 1），并用这个分类器对原始样本集 S 进行分类，将错分的多数类样本并入集合 T；

3、对于集合 T，使用 Tomek links 方法剔除多数类样本，最终得到训练样本 T。

这个基于 Tomek links 的方法称为 One-Sided Selection。算法中利用 Tomek links 方法剔除多数类中的边界点和噪声点，未被 1-NN 错分的多数类则被视为多余样本，最终得到一个较为平衡的样本集。

结果




Running time = 95.36332128395043 seconds.

Accuracy: 0.999508

Recall: 0.763514

Precision: 0.941667

Fscore: 0.843284

### 3.1.3 SMOTE 过采样方法

在大部分人看来，过采样更加具有优越性，原因很简单，欠采样删除一部分数据，从而损失了原有的数据集的部分信息；相反，过采样的方法，不仅没有删除原有样本集的信息，还生成了更多的样本数据，所以，在应用上相对更加的广泛。但是，实际上，过采样‘生成’数据的方式同样有他的局限性，因为过分的过采样会造成严重的重复数据的情况，是得少数类的样本集的方差偏低，改变了样本的分布。

SMOTE 过采样（Synthetic Minority Over-sampling Technique）[7],主要的思想来源于图像处理中的仿射不变性，即对样本做简单的旋转等仿射变换，不会改变图像的类，因此可以借此产生更多的样本。在 SMOTE 中具体为通过一些位置相近的少数类样本生成新样本达到平衡数据类别的目的。

SMOTE 的算法流程：

设置上采样的倍率 N，即每个少数类样本需要产生 N 个相应的新样本。

1， 记整个训练集为 T，少数类样本集为 T1，多数类样本集为 T2。对 T1 中的每一个样本 $x_i$ ，在训练集 T 中，通过 m-NN(kNN 的近邻数为 m)，并记其中少数类样本的数量为 $m_i$；

2， If $m_i = 0$，即 $x_i$ 附近 m 个最邻样本点都是多数类，则认为 $x_i$ 是噪声点，不操作；

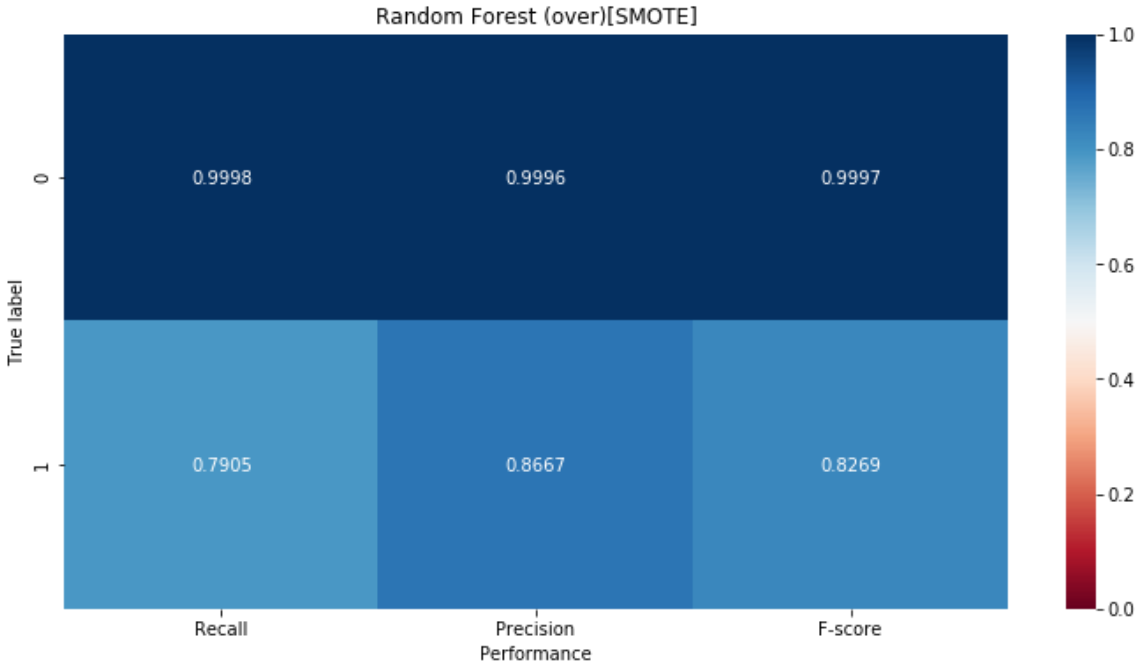
elif $m_i > \frac{m}{2}$ ，认为 $x_i$ 是安全点（非边界点），不操作；

else 认为 $x_i$ 为边界点，将其划分为危险集 D。

3， 对 D 数据集中的每一个样本点，通过 kNN(通常取 k=5)搜索得到其 k 个近邻的样本，在这些邻近样本点之间通过线性插值，生成指定数量的新的少数类样本点。

这种基于 SOMTE 的过采样算法，称为 Borderline SMOTE 算法，他的优势在于利用边界的少数类样本点，而不是所有的少数类样本点来产生新样本，这样可以更好的解决少数类样本点在边界被错分的情况，从而提升模型的效率。

结果：




Runnning time = 168.9119964444444 seconds.

Accuracy: 0.999427

Recall: 0.790541

Precision: 0.866667

Fscore: 0.826855

### 3.1.4 综合采样

前面讲过两类采样方法：欠采样和过采样，但都只是针对一类样本集进行处理，对多数类样本欠采样，对少数类样本过采样，而且各自都有其局限性。

这里再介绍一种综合采样法，结合了欠采样和过采样方法。

**SMOTE + Tomek links [7]**

算法流程：

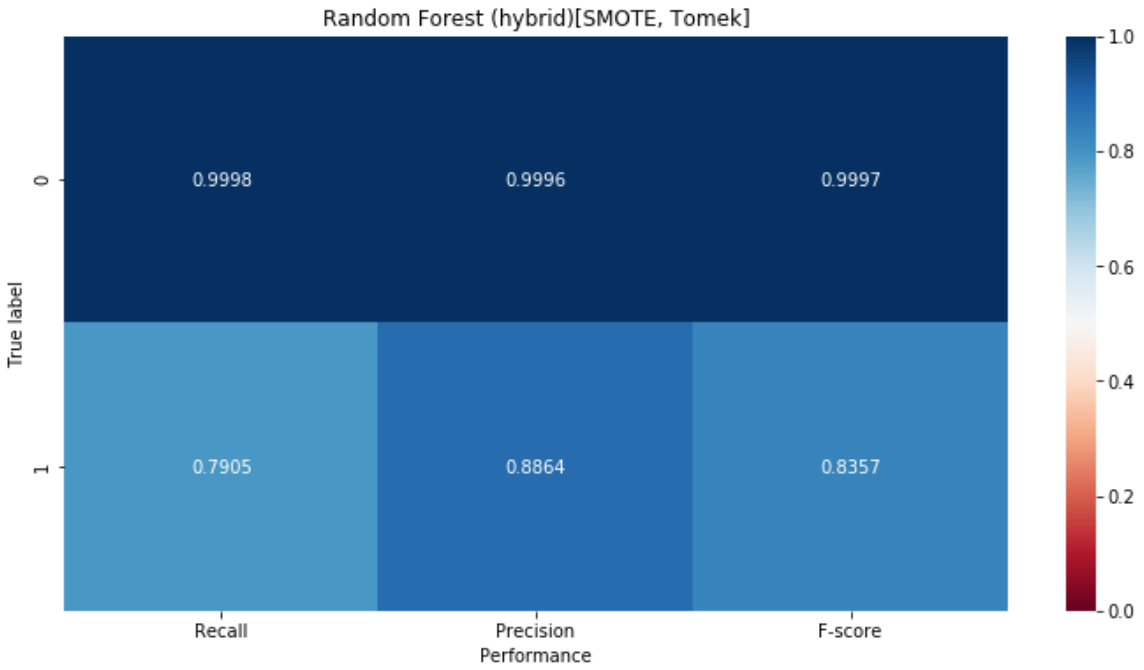
1， 利用 SMOTE 法生成新的少数类样本集，得到扩充后的数据集 T；

2， 利用 Tomek links 方法，找到 T 中的 Tomek links 对；

3， 剔除 T 中的 Tomek links 对，得到新的平衡样本集。

这种混合采样的优势在于：通常的 SMOTE 过采样在平衡数据集的同时，也扩展了少数类样本的样本空间（即部分新生成的少数类样本点位于多数类样本的空间内），随之而来的是更多的噪声点或者混乱的边界。混合采样，在 SMOTE 采样之后，又通过 Tomek links 剔除掉噪声点或混乱的边界点，因为第一步的 SMOTE 已经平衡了数据集，而 Tomek links 又很好的解决了噪声和边界的问题，这样使得新的数据集的分类效果更好。

结果




Running time: 175.3319360000005 seconds.

Accuracy: 0.999462

Recall: 0.790541

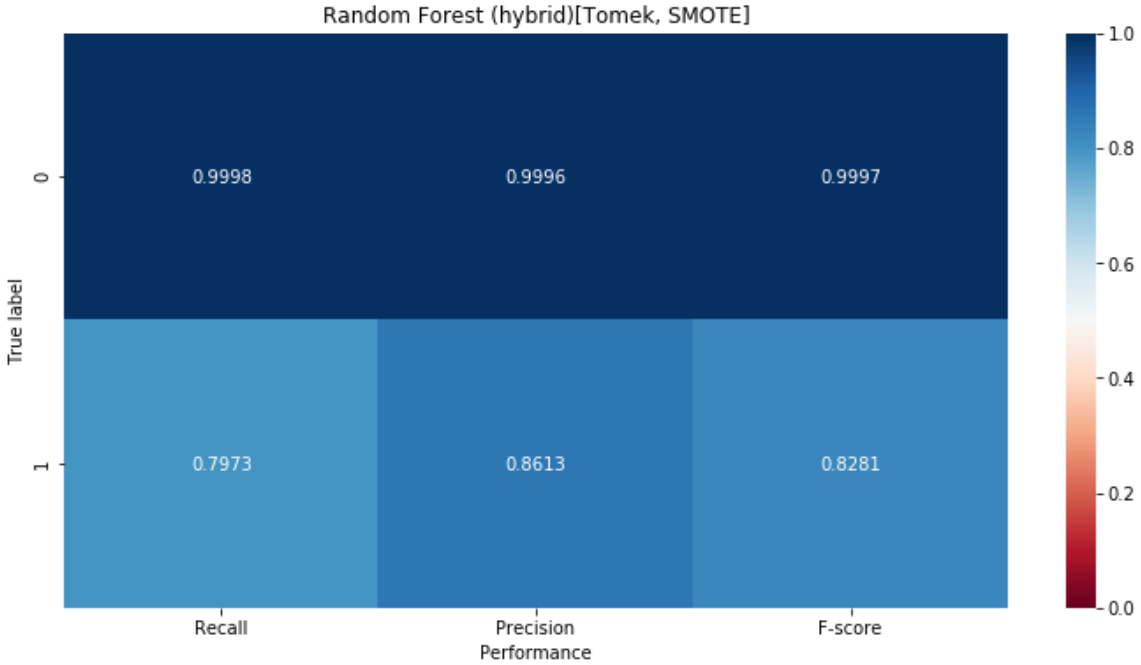
Precision: 0.886364

Fscore: 0.835714

**Tomek links + SMOTE [7]**

调换欠采样和过采样的顺序，所得到的混合采样的结果是不一样的。前面的混合采样方法是先生成新的少数类样本，再剔除 Tomek links 对，多数类样本的总体基本没变，只是生成了多的少数类，并剔除了多数类中新产生的少数类噪声点；

而先用欠采样方法，删除一定数量的多数类样本，使总体总体样本量保持一定的比例，再通过 SMOTE 的方法，产生另一部分新的少数类样本，从而是整体样本集达到较为平衡的效果。




Running time = 170.1840703209882 seconds.

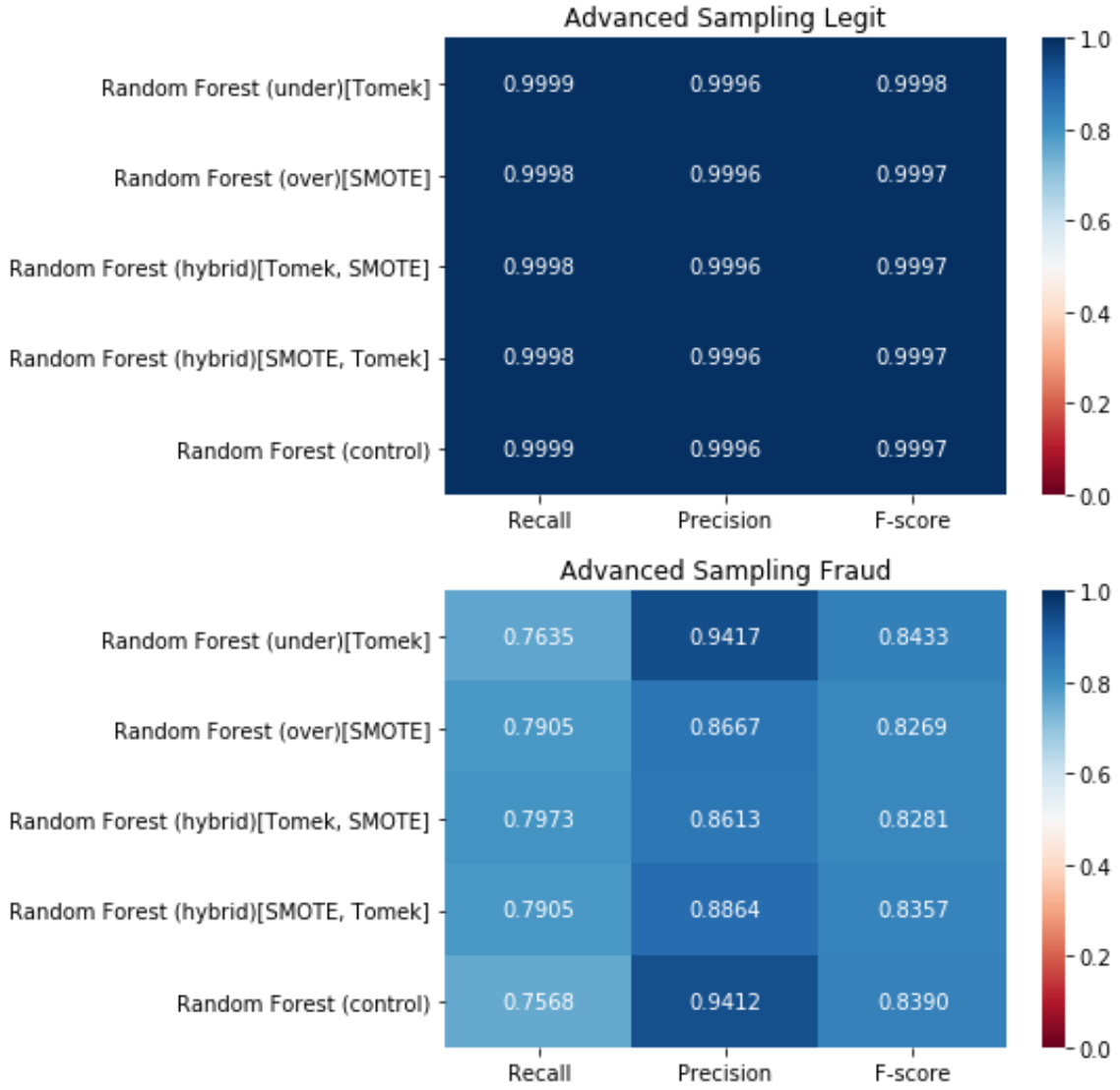
Accuracy: 0.999427

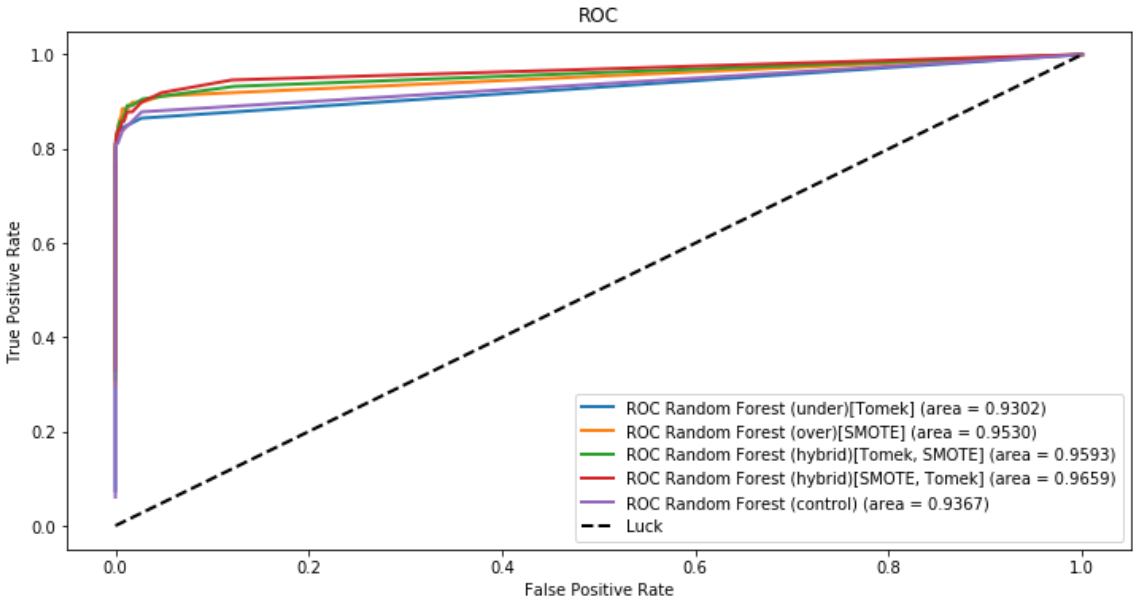
Recall: 0.797297

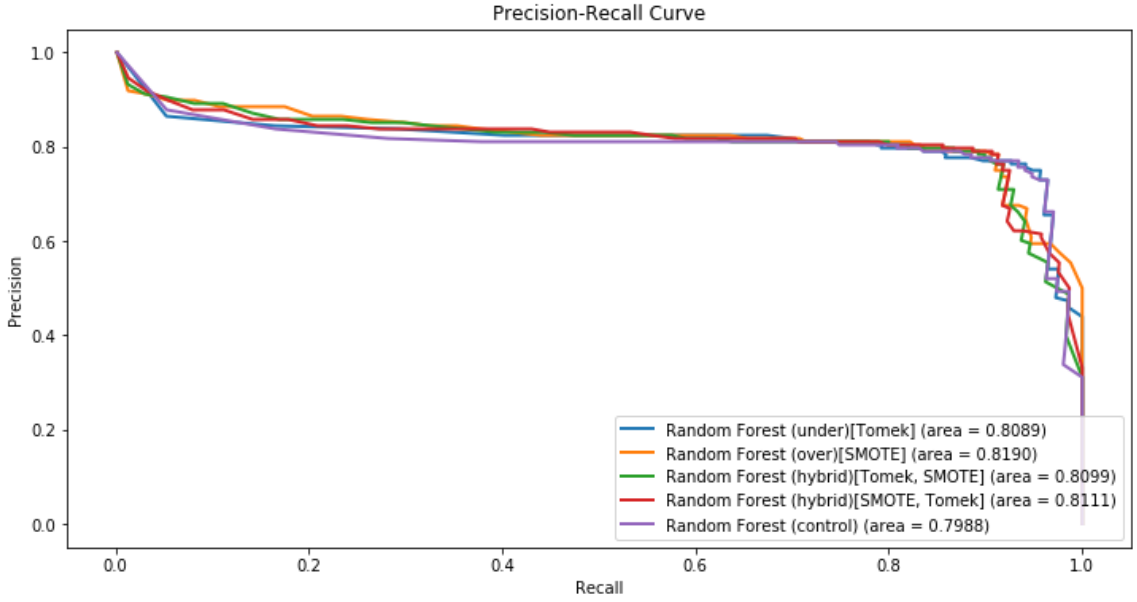
Precision: 0.861314

Fscore: 0.828070

最后，将各种采样方法作了一个综合比较：






## 3.2 特征选择

### 3.2.1 基于正态分布的特征选择

其原理很简单：先取数据集中的正常交易，分析其均值和方差，然后用这个均值和标准差差去定义标准特征集的分布；然后预测，利用每笔交易的各个特征数据去计算该笔交易发生的概率，设定接受域和阈值，低于设定阈值的交易都被认为是诈骗交易。这个方法的前提是数据符合正态分布且不同类别标签的特征有明确的差异。

具体描述为[3]：

假设 n 维（本例中的维度为 30）的数据集合形如 $\vec{x}_i = (x_{i,1}, ..., x_{i,n}), i \in \{1, ..., m\}$，那么可以计算每个维度的均值和方差 $\mu_j, \sigma_j, j \in \{1, ..., n\}$. 具体来说，于 $j \in \{1, ..., n\}$，可以计算：

$$\mu_j = \sum_{i=1}^{m} x_{i,j}/m$$

$$\sigma_j^2 = \sum_{i=1}^{m}(x_{i,j} - \mu_j)^2/m$$

在正态分布的假设下，如果有一个新的数据 $\vec{x}$，可以计算概率 $p(\vec{x})$如下：

$$p(\vec{x}) = \prod_{j=1}^{n} p(x_j; \mu_j, \sigma_j^2) = \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_j} \exp(-\frac{(x_j-\mu_j)^2}{2\sigma_j^2})$$

根据概率值的大小就可以判断 x 是否属于异常值（非正常交易）。

先简单对数据做了描述性的统计，得到如下结果：

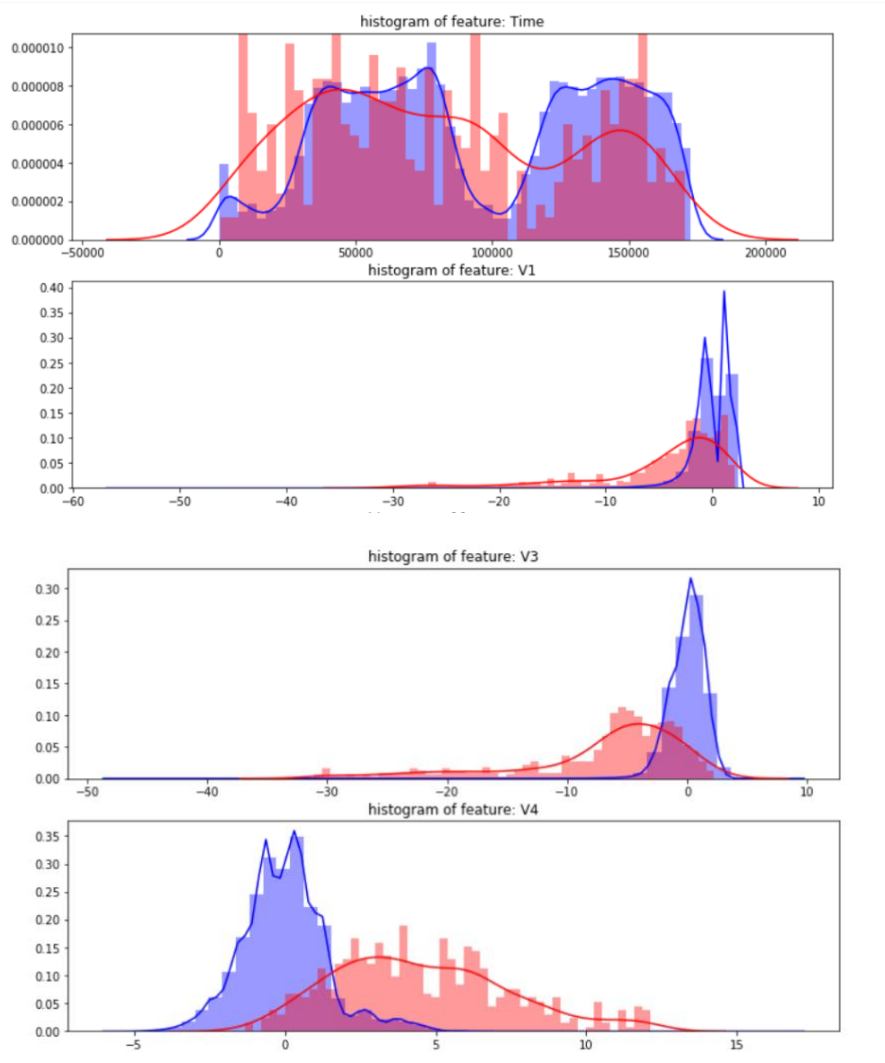



图 1 数据的描述统计

我们画出了两个类别在各个特征上的直方图，来观察正常交易（蓝色）和诈骗交易（红色）在不同的特征上是否有显著的区别。一共有 30 组直方图，这里仅以 4 个为例简单说明：

观察得到的信息是，1、除时间变量外，其他的特征的分布都近似的服从正态分布；2、两个不同类的数据分布上还是有着一定的区别。

由此，我们试着去由两类样本间的 T 检验来筛选“好的”特征变量，然后，再通过正态分布的特性去检测异常值（诈骗交易）。

然后，计算出标准集的分布：mean：4.495588e-03，std ： 1.077934e-02

为了测试这个基于正态分布的异常值检测的效果，我们把部分正类（合法交易）和负类（诈骗交易）的数据代入测试，得到了一个关于正类和负类在设定不同阈值的情况下，其 Recall 和 Precision 的值。具体为：

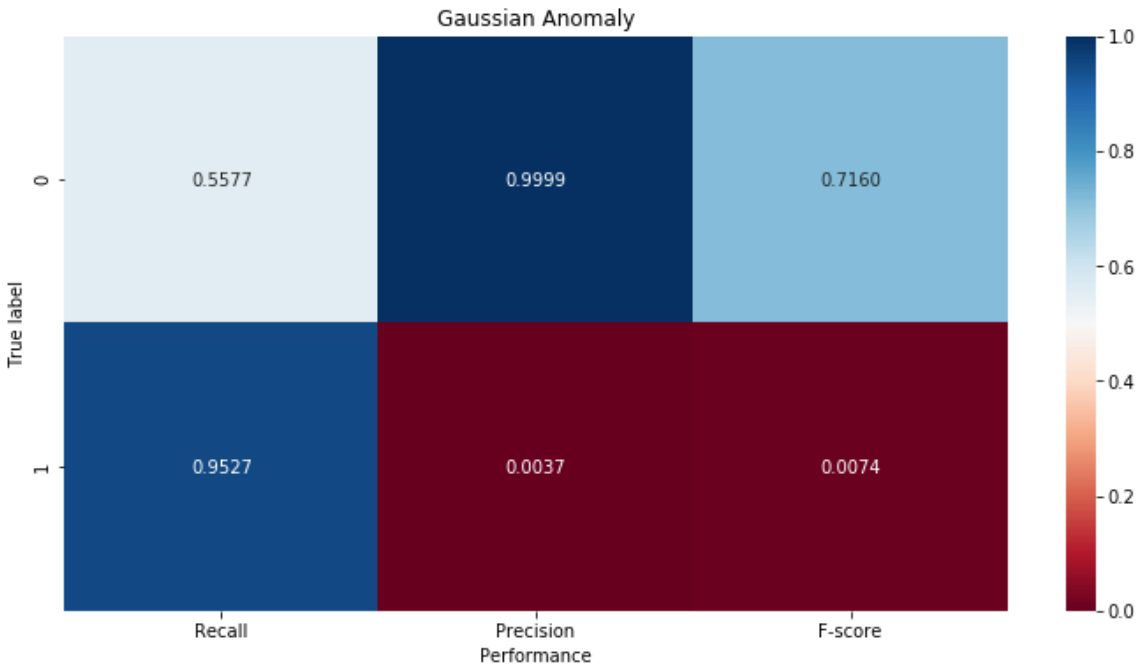
表 1 不同阈值下正常交易和诈骗交易的 Recall

| 正常交易 | | | | | 诈骗交易 | | | |
|---|---|---|---|---|---|---|---|---|
| 1.00% | 4.33E-40 | 72.00% | 2.29E-03 | | 1.00% | 0.00E+00 | 72.00% | 1.11E-32 |
| 2.00% | 7.89E-23 | 73.00% | 2.54E-03 | | 2.00% | 0.00E+00 | 73.00% | 5.53E-32 |
| 3.00% | 1.62E-16 | 74.00% | 2.81E-03 | | 3.00% | 0.00E+00 | 74.00% | 2.27E-31 |
| 4.00% | 1.06E-13 | 75.00% | 3.11E-03 | | 4.00% | 0.00E+00 | 75.00% | 7.40E-31 |
| 5.00% | 4.02E-12 | 76.00% | 3.44E-03 | | 5.00% | 0.00E+00 | 76.00% | 6.60E-30 |
| 6.00% | 3.87E-11 | 77.00% | 3.80E-03 | | 6.00% | 0.00E+00 | 77.00% | 5.01E-29 |
| 7.00% | 2.22E-10 | 78.00% | 4.18E-03 | | 7.00% | 0.00E+00 | 78.00% | 2.39E-27 |
| 8.00% | 8.45E-10 | 79.00% | 4.62E-03 | | 8.00% | 0.00E+00 | 79.00% | 1.04E-25 |
| 9.00% | 2.64E-09 | 80.00% | 5.12E-03 | | 9.00% | 0.00E+00 | 80.00% | 3.44E-22 |
| 10.00% | 6.78E-09 | 81.00% | 5.66E-03 | | 10.00% | 0.00E+00 | 81.00% | 1.75E-19 |
| 11.00% | 1.44E-08 | 82.00% | 6.27E-03 | | 11.00% | 0.00E+00 | 82.00% | 5.85E-18 |
| 12.00% | 2.83E-08 | 83.00% | 6.95E-03 | | 12.00% | 0.00E+00 | 83.00% | 1.23E-17 |
| 13.00% | 5.00E-08 | 84.00% | 7.68E-03 | | 13.00% | 0.00E+00 | 84.00% | 2.53E-17 |
| 14.00% | 8.34E-08 | 85.00% | 8.53E-03 | | 14.00% | 0.00E+00 | 85.00% | 1.73E-13 |
| 15.00% | 1.30E-07 | 86.00% | 9.45E-03 | | 15.00% | 0.00E+00 | 86.00% | 6.22E-12 |
| 16.00% | 1.97E-07 | 87.00% | 1.05E-02 | | 16.00% | 0.00E+00 | 87.00% | 1.44E-11 |
| 17.00% | 2.88E-07 | 88.00% | 1.16E-02 | | 17.00% | 0.00E+00 | 88.00% | 3.09E-10 |
| 18.00% | 4.02E-07 | 89.00% | 1.28E-02 | | 18.00% | 0.00E+00 | 89.00% | 1.91E-08 |
| 19.00% | 5.61E-07 | 90.00% | 1.42E-02 | | 19.00% | 0.00E+00 | 90.00% | 3.79E-08 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 20.00% | 7.56E-07 | 91.00% | 1.58E-02 | | 20.00% | 0.00E+00 | 91.00% | 1.52E-07 |
| 21.00% | 1.01E-06 | 92.00% | 1.76E-02 | | 21.00% | 0.00E+00 | 92.00% | 1.75E-06 |
| 22.00% | 1.33E-06 | 93.00% | 1.98E-02 | | 22.00% | 0.00E+00 | 93.00% | 4.64E-06 |
| 23.00% | 1.72E-06 | 94.00% | 2.24E-02 | | 23.00% | 0.00E+00 | 94.00% | 2.79E-05 |
| 24.00% | 2.25E-06 | 95.00% | 2.54E-02 | | 24.00% | 2.37e-313 | 95.00% | 3.25E-05 |
| 25.00% | 2.88E-06 | 96.00% | 2.93E-02 | | 25.00% | 2.32E-199 | 96.00% | 1.20E-04 |
| 26.00% | 3.59E-06 | 97.00% | 3.45E-02 | | 26.00% | 4.42E-186 | 97.00% | 3.60E-04 |
| | ... | 98.00% | 4.24E-02 | | | ... | 98.00% | 6.66E-04 |
| 71.00% | 2.06E-03 | 99.00% | 5.52E-02 | | 71.00% | 4.43E-33 | 99.00% | 1.45E-03 |

为了更直观的显示器分类效果，我们取了不同的阈值来比较了其回收率和精确率。如下图：

两个取得阈值分别为 0.001 和 7.883e-30。



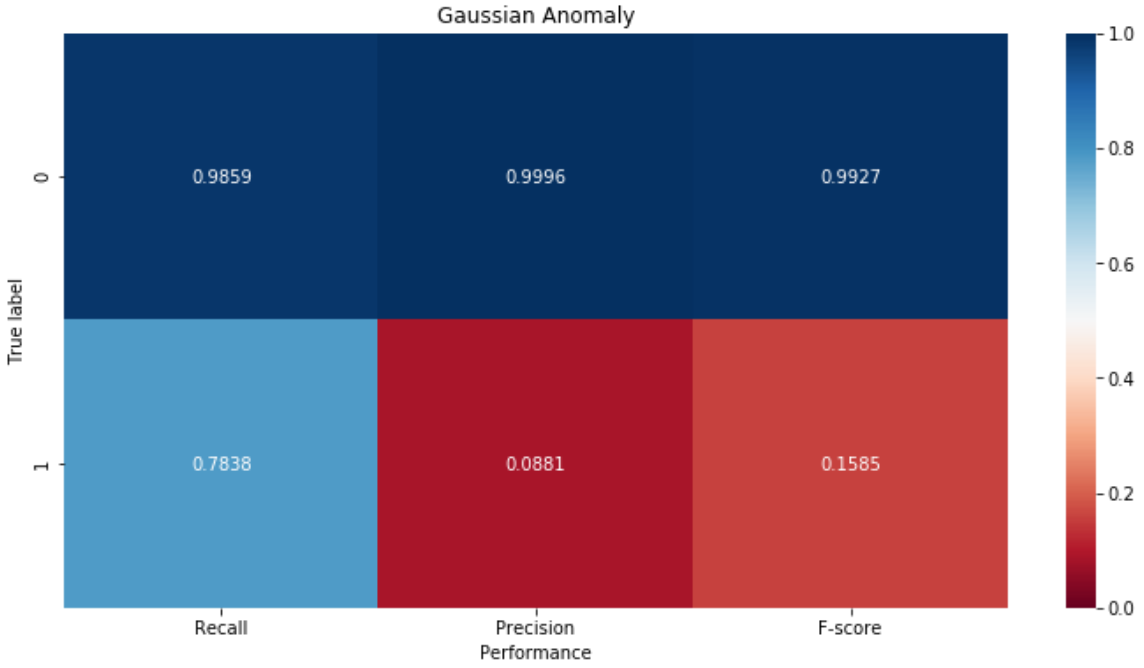



图 2 正态检验的不同阈值结果

可以看到，阈值越大，诈骗交易的 Recall 就越高，同时正类的 Recall 就急剧下降；而当正类的 Recall 提高时，负类的 Recall 又相应的下降。

直观上去理解，就是我们对于诈骗类交易的定义越严格，阈值就越小，相应的正类被错分的机会就越小，即正类的 Recall 就相对较高；如果，我们对诈骗类交易的定义更宽泛，阈值就越大，则将有更多的样本被划分为负类，相应的负类的 Recall 就越高。

但是，注意到：随着诈骗交易的 Recall 的提高，相应的更多的正例被也被误分为诈骗交易，两个相互冲突。没有办法达到最优。

### 3.2.2 基于自动编码器的特征提取

关于自动编码器（Auto Encoder）：

AE 是一种无监督的学习方式，不同于其他的深度学习的方式，自编码的学习过程不需要带标签的数据，其训练的 Loss 也是由输入的特征和输出的特征之间的差异决定，其目标是通过使用比输入节点更少的隐藏节点（在编码器一端）预测输入（训练该网络使其输出尽可能与输入相似），为此该网络需要尽可能多地将信息编码到隐藏节点中。

具体结构如下：

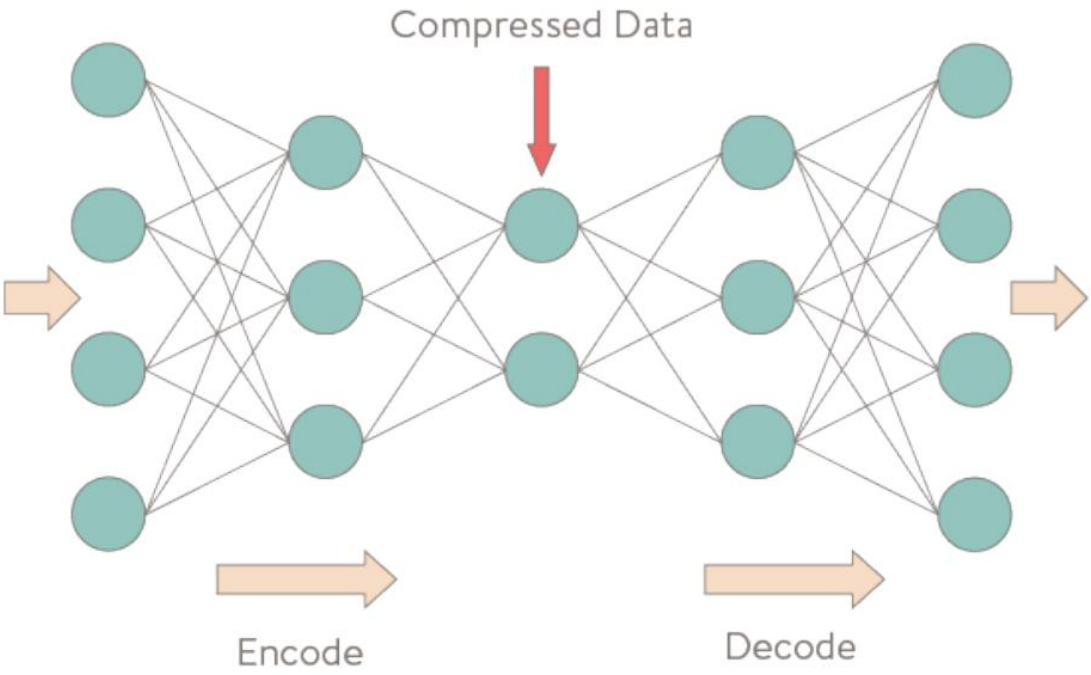
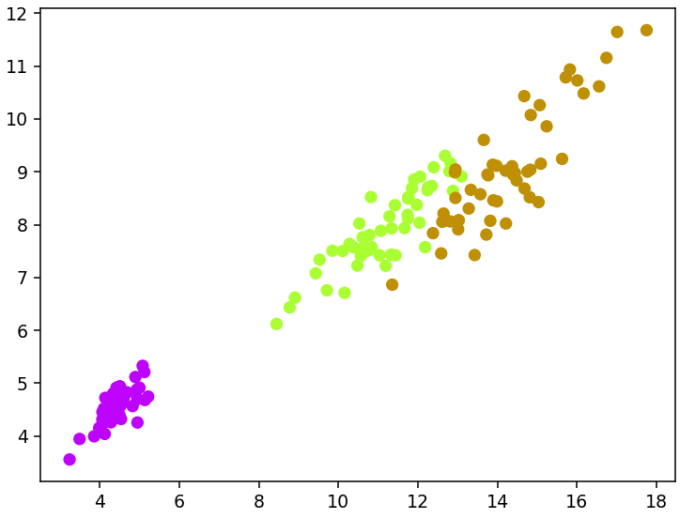



图 3 自动编码器

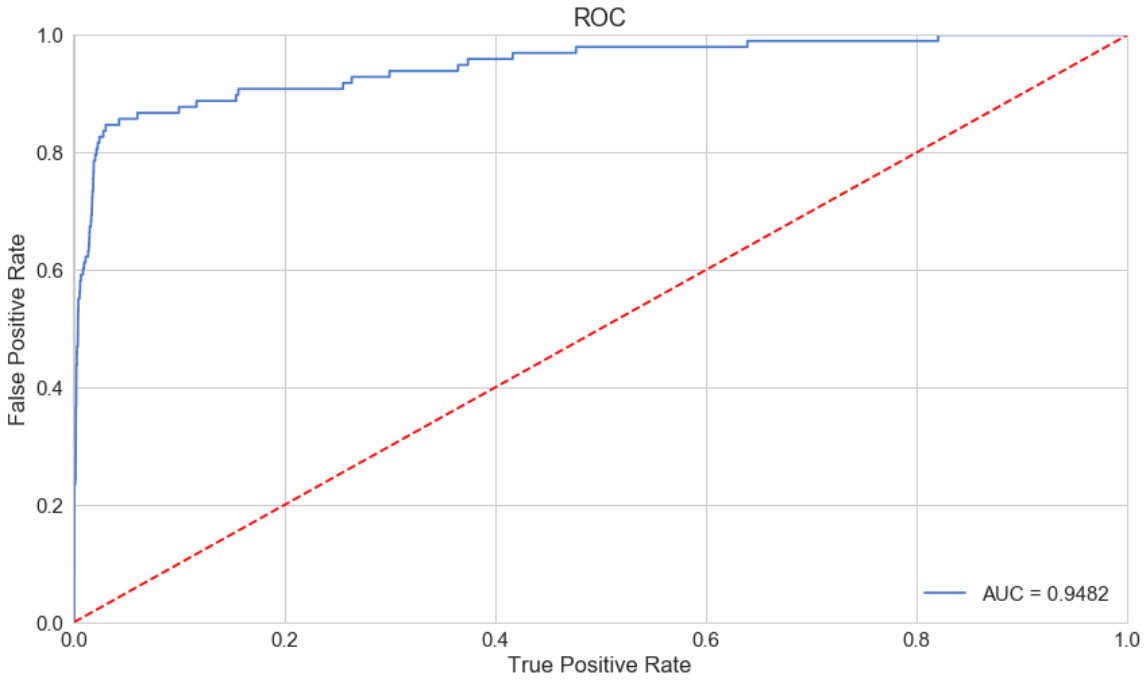
先对原始数据进行编码（压缩），再对压缩后得到的特征进行解码，最终得到与输入变量 X 维度相同的新的特征变量 X’,而在原始数据集中的难以用来区分的特征，在新的特征空间中就可能具有很好的分类性质。以鸢尾花的数据集为例，下图为我们经过 AE 编码后的特征图（batch_size=50, n_epochs=1000），在新的特征空间中就可以很容易的利用一些简单分类器进行分类。

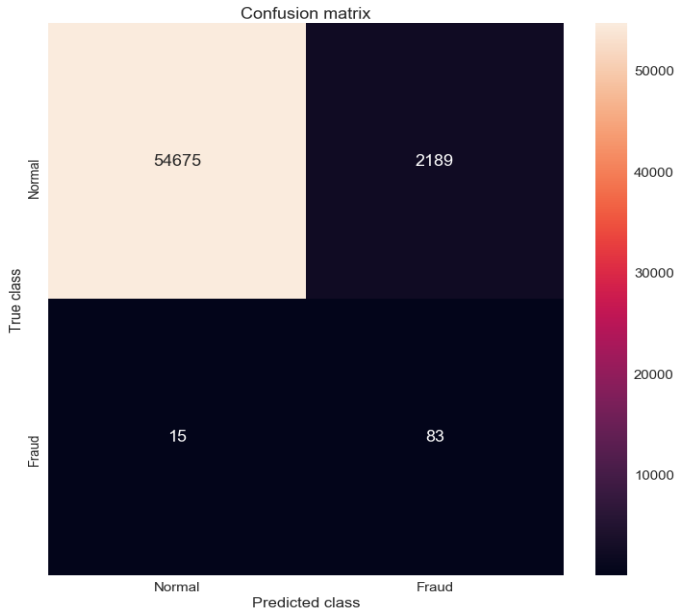


这里也用类似的想法，去处理银行交易数据的变量，提取新的特征变量，再通过分类去判别该交易是否属于诈骗交易。

模型参数：2*encoder+2*decoder

通过实验，得到如下结果：




得到的具体数据为：

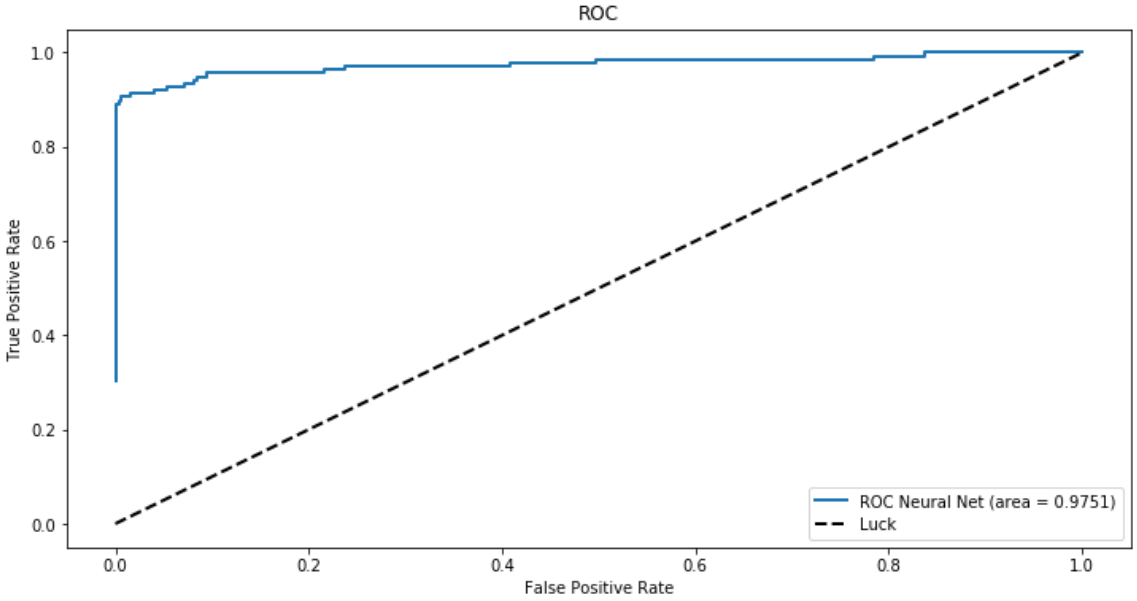
| | Recall | Precision | F |
|---|---|---|---|
| 0(normal) | 0.9615 | 0.9997 | 0.9802 |
| 1(fraud) | 0.8469 | 0.0365 | 0.5102 |

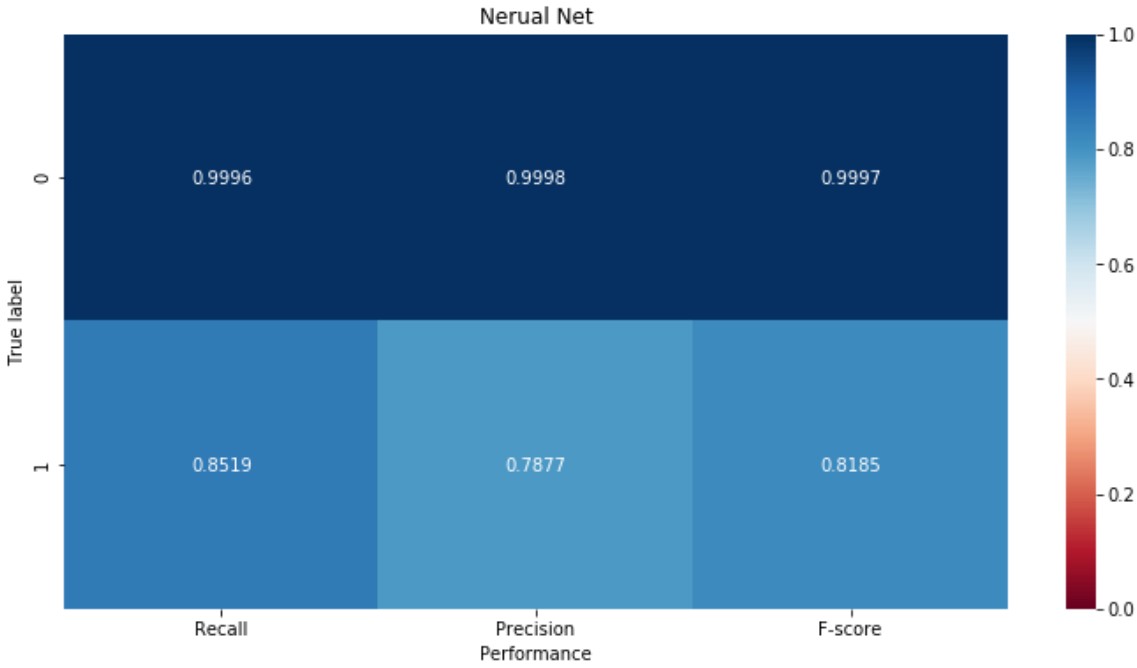
Running time: 1639.4175466666666 seconds.

### 3.2.3 基于 MLP 的特征提取和分类

神经网络也可以用于特征提取和分类，本文使用的就是最简单的多层网络，简单来看，就是前文自编码的编码部分，不同的是 MLP 是有监督学习。后来发现，其有监督的特点，使得其在各项指标上较自动编码器有较大提升。

模型参数：hidden_layer_sizes=(200, 100,)




Running time: 166.27167683950393 seconds.

Accuracy: 0.999403

Recall: 0.851852
Precision: 0.787671
Fscore: 0.818505

## 4. 结论

我们将所有的不平衡数据的处理方法的效果整理如下：

表 2. 正常交易在不同方法中的分类效果(测试集)

| For normal | | Recall | Precision | F-score | Time |
|---|---|---|---|---|---|
| 重采样法 | Tomek links | 0.9999 | 0.9996 | 0.9998 | 95.36s |
| | SMOTE | 0.9998 | 0.9996 | 0.9997 | 168.91s |
| | SMOTE + Tomek links | 0.9998 | 0.9996 | 0.9997 | 175.33s |
| | Tomek links + SMOTE | 0.9998 | 0.9996 | 0.9997 | 170.18s |
| | Control | 0.9999 | 0.9996 | 0.9997 | 79.99s |
| 特征选择 | Gaussian | 0.9859 | 0.9996 | 0.9927 | 0.82s |
| | AutoEndcoder | 0.9615 | 0.9997 | 0.9802 | 1639.41s |
| | MLP | 0.9996 | 0.9998 | 0.9997 | 166.27s |

表 3. 诈骗交易在不同方法中的分类效果(测试集)

| For Fraud | | Recall | Precision | F-score | Time |
|---|---|---|---|---|---|
| 重采样法 | Tomek links | 0.7635 | 0.9417 | 0.8433 | 95.36s |
| | SMOTE | 0.7905 | 0.8667 | 0.8269 | 168.91s |
| | SMOTE + Tomek links | 0.7905 | 0.8864 | 0.8357 | 175.33s |
| | Tomek links + SMOTE | 0.7973 | 0.8613 | 0.8281 | 170.18s |
| | Control | 0.7568 | 0.9412 | 0.839 | 79.99s |
| 特征选择 | Gaussian | 0.7838 | 0.0881 | 0.1585 | 0.82s |
| | AutoEndcoder | 0.8469 | 0.0365 | 0.5102 | 1639.41s |
| | MLP | 0.8519 | 0.7877 | 0.8185 | 166.27s |

得到的结论如下：

1. 就重采样方法来说，有重采样的效果要比无重采样的效果好，但各种采样方法相比较起来，几乎没有太大差别；

2. 对于特征选择的方法来说，基于正态分布的特征选择和自动编码器，可以再回

收率上表现的很好，但在精度上就表现的很差；相对来说 MLP 在各方面表现的都很好；

3. 在数据集严重不平衡时，重采样的效果并不明显，相反在特征选择方面入手，效果好很多。

Reference

[1] http://bbs.tianya.cn/post-develop-1424282-1.shtml

[2] Farquad, M. A. H., and Indranil Bose. "Preprocessing unbalanced data using support vector machine." Decision Support Systems 53.1 (2012): 226-233.

[3] Han, Hui, Wen-Yuan Wang, and Bing-Huan Mao. "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning." Advances in intelligent computing (2005): 878-887.

[4] Chawla, Nitesh V., et al. "SMOTE: synthetic minority over-sampling technique." Journal of artificial intelligence research 16 (2002): 321-357.

[5]https://www.researchgate.net/publication/230766603_How_Many_Trees_in_a_Random_Forest

[6] https://www.jair.org/media/953/live-953-2037-jair.pdf

[7] http://pages.stern.nyu.edu/~fprovost/Papers/skew.PDF