# 异常检测–信用卡欺诈分析 数据挖掘课程项...

## 异常检测–信用卡欺诈分析 数据挖掘课程项目报告

项目代码仓库地址：https://github.com/byuegv/credit-card-fraud-analysis

### 成员及分工

| 姓名 | 学号 | 完成的主要工作 |
| --- | --- | --- |
| 赵一诺 | 3120191078 | 关联分析 |
| 葛晶 | 3120190991 | 不平衡数据探索 |
| 李世林 | 3120191017 | 数据预处理及分类建模 |
| 谢斌辉 | 3120191059 | 数据可视化 |
| 吴楠楠 | 3220190895 | 项目文档(PPT) |

### 项目说明

#### 1. 项目背景及分析

> 在数据挖掘中，异常检测（也称为离群值检测）是对罕见项目，事件或观察结果的识别，这些事务与大多数数据有显着差异，从而引起怀疑。异常检测是一种用于识别不符合预期行为的异常模式的技术，称为异常值。异常值检测在在商业中有许多应用，例如:入侵检测（识别可能表明黑客入侵的网络流量中的异常模式）、系统健康监控（在MRI扫描中发现恶性肿瘤）以及检测信用卡交易中的欺诈。

#### 2. 问题描述

> 信用卡公司能够识别欺诈性的信用卡交易非常重要，在本项目中，我们将以信用卡欺诈检测为案例进行研究。采用基于机器学习的方式对数据进行分类建模，然后，使用该模型来识别新交易是否为欺诈行为。

#### 3. 数据集描述

本项目使用Kaggle的信用卡欺诈检测的数据集：Credit Card Fraud Detection：

- 该数据集包含2013年9月欧洲持卡人通过信用卡进行的交易。此数据集显示了两天内发生的交易，在284,807笔交易中有492起欺诈。
- 数据集高度不平衡，阳性类别（欺诈）仅占所有交易的0.172%。
- 数据集仅包含数字输入变量，它们是PCA转换的结果。其中包含来自28个“主成分分析（PCA）”转换特征的数值，即V1至V28。 此外，由于机密性问题，我们无法得到相关数据的原始特征和更多背景信息。

- 特征值 `Time` 和 `Amount` 尚未经过PCA转化。 `Time` 包含数据集中每个事务和第一个事务之间经过的秒数。 `Amount` 是交易金额，此特征可以用于与示例相关的成本敏感型学习。
- 特征值 `Class` 是响应变量，在发生欺诈时其值为1，否则为0。
- 数据集中没有缺失值。

### 4. 算法/模型

#### 4.1 分类建模

> 分别采用有监督的方法（MLP，LogisticRegression）和无监督方法（SVM，LOF，Isolation Forest，KNN）Credit Card Fraud Detection进行分类建模。

#### 4.2 关联分析

> 采用FP-growth算法对属性V1—V28, Amount, Time以及是否发生欺诈进行关联分析，找出所有大于最小支持度的频繁项集，然后由频繁项集产生关联规则。

### 5. 评估指标

- 对分类建模算法/模型主要应用精确率、召回率以及F1值进行评估。
- 关联规则挖掘则在一定支持度范围内进行评估。

## 数据预处理

Credit Card Fraud Detection 数据集一共包含31列数据，其中Time,Amount,V1-V28为特征列，Class列表示是否为欺诈交易，值为1时为欺诈交易，值为0时为正常交易。
此数据集共有284807条交易记录，无缺失值，并且所有数据均为数值类型。
经过分析，我们首先将Time列的数据修正到24小时内，然后对Time和Amount进行归一化处理，其他列的数据保持不变。

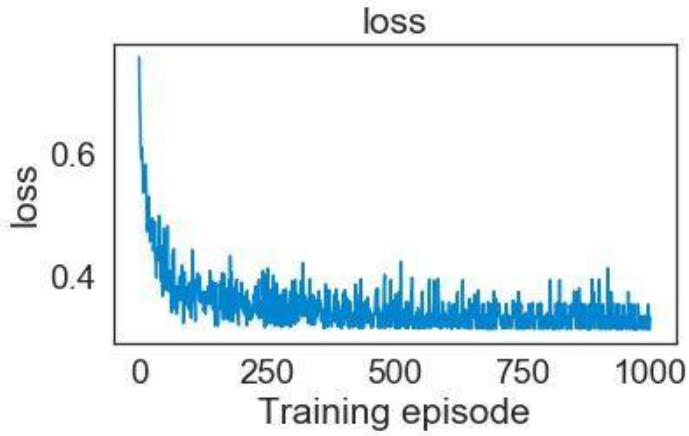
## 结果与分析

### 1. 分类建模

使用完整的Credit Card Fraud Detection数据集，按照训练数据：测试数据=0.8:0.2划分训练集和测试集，并分别在有监督和无监督方法下进行训练和测试，各算法/模型的精确率，召回率及F1值和预测准确率如下所示。

#### 使用MLP进行分类建模

在这个部分，我们使用的是经过下采样后的平衡数据集。首先，按照9:1的比例划分训练集与测试集。其次，搭建一个2层的简易神经网络，并使用交叉熵函数为损失函数。
定义训练的轮数为1000轮，每次抽取50个样本进行训练，得到的训练损失如下：



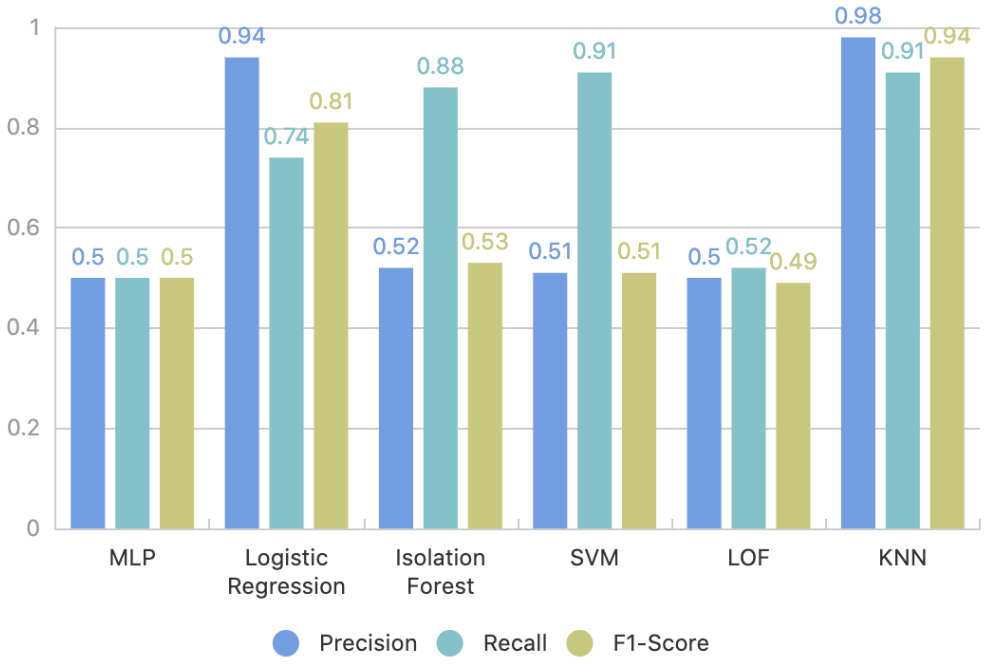

可以看到，损失已经降到了一个非常低的水平，网络趋于稳定。然后在测试集上进行测试，得到分类的正确率为95.9%。可以看出，使用MLP方法进行分类建模，能够很好的识别出欺诈数据。

- 有监督方法

| Algorithm/ Model | Class | Precision | Recall | F1-Score | Prediction Accuracy |
|---|---|---|---|---|---|
| MLP | Normal | 1.0 | 1.0 | 1.0 | 99.80% |
| | Fraud | 0.0 | 0.0 | 0.0 | |
| Logistic Regression | Normal | 1.0 | 1.0 | 1.0 | 99.89% |
| | Fraud | 0.89 | 0.47 | 0.62 | |

- 无监督方法

| Algorithm/ Model | Class | Precision | Recall | F1-Score | Prediction Accuracy |
|---|---|---|---|---|---|
| Isolation Forest | Normal | 1.0 | 0.96 | 0.98 | 95.95% |
| | Fraud | 0.04 | 0.81 | 0.06 | |
| SVM | Normal | 1.0 | 0.94 | 0.97 | 94.26% |
| | Fraud | 0.03 | 0.89 | 0.05 | |
| LOF | Normal | 1.0 | 0.96 | 0.98 | 96.06% |
| | Fraud | 0.0 | 0.08 | 0.01 | |
| KNN | Normal | 1.0 | 1.0 | 1.0 | 99.97% |
| | Fraud | 0.97 | 0.83 | 0.89 | |

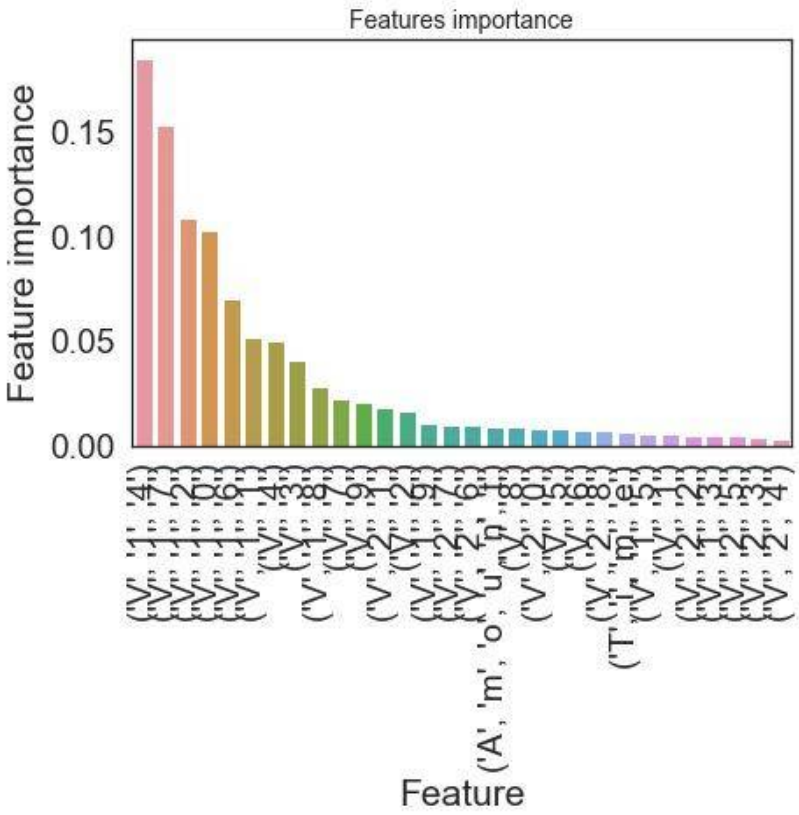



根据实验结果，在这个数据集上有监督方法Logistic Regression以及无监督方法KNN表现较好，不仅具有更高的预测准确率，同时对Fraud（欺诈交易）有更高的精确率，召回率和F1-Score。

## 2. 关联分析

### 使用Random Forest算法挖掘重要属性

因为未知特征V1-V28过多，所以要选择重要特征进行规则分析。可以使用Random Forest算法对属性的重要性进行排序。将熵作为训练标准，将随机状态的数量设置为42，将最大深度的数量设置为10，将估计器的数量设置为100，将最大特征设置为自动。然后使用模型可视化各个特征的重要性。结果如下图所示：




从上图可以看出，特征V14，V17，V12，V10，V16，V11对于属性Class的重要性最高。因此选用这些特征和Time，Amount属性一起，进行频繁项集和关联规则挖掘。

### 使用KNN算法对属性值预处理

由于属性值都是连续值，无法直接进行频繁项集挖掘，所以首先使用KNN算法，把各个属性进行聚类，达到离散化的目的。把特征V14，V17，V12，V10，V16，V11，Amount均聚成10类。特征Time聚成24类，因为一天24小时。特征Class不变。

在关联规则的挖掘部分，如果使用不平衡的数据集，则无法挖掘到和异常样本相关的规则（因为正常样本所占的比例过大，导致支持度很大），所以我们使用平衡数据及，挖掘和异常值相关的规则。

### 使用FP-growth算法进行频繁项集挖掘

相比于经典的Apriori算法，FP-Growth算法更进一步，通过将交易数据巧妙的构建出一颗FP树，然后在FP树中递归的对频繁项进行挖掘。FP-Growth算法仅仅需要两次扫描数据库，第一次是统计每个商品的频次，用于剔除不满足最低支持度的商品，然后排序得到FreqItems。第二次，扫描数据库构建FP树。在这里，我们设定频繁项集的支持度为100，挖掘到所有含有类别（class_0或class_1）的频繁项集（仅显示部分，全部频繁项集请见代码）。

```
frequent pattern:  ('class_1', 'v11_8') , support: 101
frequent pattern:  ('class_1', 'v14_4') , support: 103
frequent pattern:  ('class_1', 'v10_2') , support: 119
...
frequent pattern:  ('amount_0', 'class_1') , support: 309
```

根据FP-growth算法，输出所有包含类别属性项的支持度在100以上的频繁项集。比如最后一项('amount_0','class_1')代表了amount_0（Amount属性经过KNN聚类的第0类）和'class_1'共同出现的频数是309。

### 导出关联规则

设定关联规则的置信概率为0.9，使用FP-growth算法导出所有和类别相关的关联规则如下（仅显示部分，全部规则请见代码）：

```
rules:  ('v11_8',)  ==> ('class_1',) , probability:  1.0
rules:  ('v14_4',)  ==> ('class_1',) , probability:  0.98
rules:  ('v10_2',)  ==> ('class_1',) , probability:  0.99
rules:  ('amount_0', 'v12_2', 'v14_5')  ==> ('class_0',) , probability:
0.98
...
rules:  ('amount_0', 'v10_6', 'v17_4')  ==> ('class_0',) , probability:
0.9675675675675676
```

输出在0.9的置信度下，所有和异常记录相关的规则。从导出的第一条规则可以看出特征V11的第8类的情况下，能够推导出这条记录是异常记录的概率是1。这是一条概率非常强的规则，我们几乎可以认定V11属性的第8类对应的是欺诈。同理，从第2，3条规则，我们也可以看出特征V14的第4类和V10的第2类，同样可以在很高的概率下推导出这是一条异常记录。

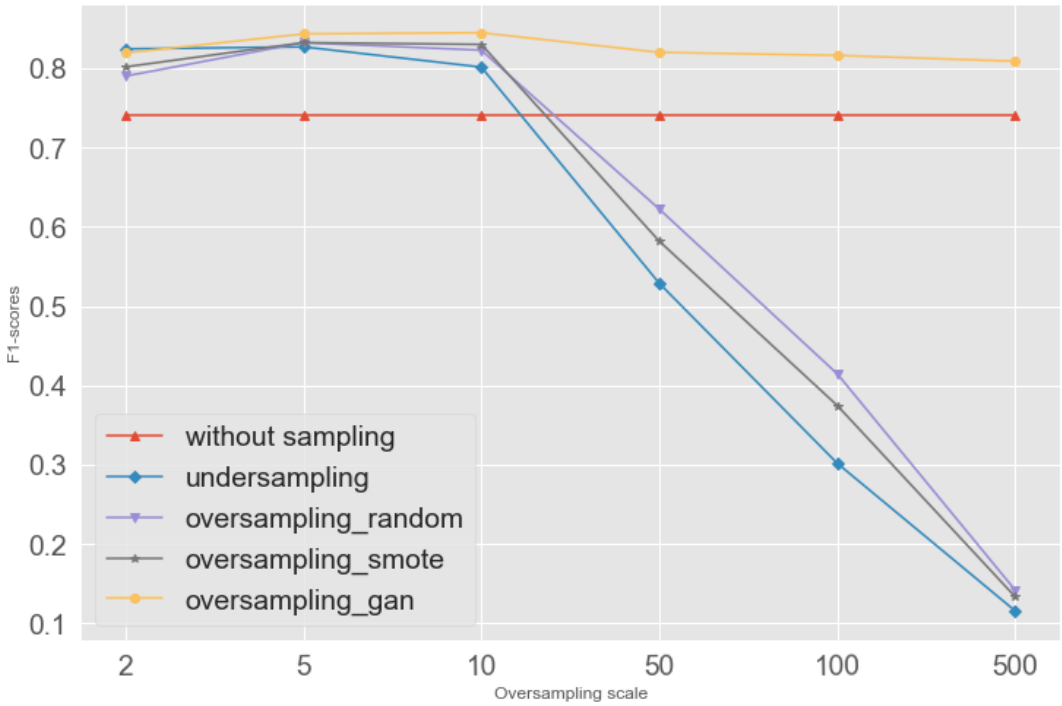
从导出的规则('amount_0', 'v12_2', 'v14_5') ==> ('class_0')可以看出，当取款的范围属于Amount的第0类的类别中，且特征V12属于第2类，特征V14属于第5类时，该条记录有98%的可能是正常记录。说明Amount属性的第0类和特征V12的第2类、V14的第5类和正常记录密切相关。由于隐私原因，无法知道特征V1-V28的具体指标，所以在此不再做详细分析。

但是通过关联规则分析，可知信用卡交易的时间与信用卡诈骗之间的关系并不是很强烈，没有导出任何取款时间和信用卡诈骗之间的规则。出现这种情况可能有三个原因，第一个就是样本不足，仅仅两天的时间无法统计出哪个时段的取款有很高可能是诈骗。第二个是取款时间和信用卡诈骗之间确实没有联系，诈骗人可能在任何一个时段取款。第三个就是在分析中，信用卡时段的划分太细，把一天的时间分为了24类，而其他的特征都是10类，因此没有导出信用卡诈骗和取款时间之间的规则。

需要注意的是，我们并没有添加Lift评价和Jaccard评估，这是因为我们发现规则挖掘的分析思路并不适合于这个数据集。因为在这个数据集中，一方面属性是连续的，虽然我们使用了KNN算法进行聚类然后对属性值进行离散化，但是这种离散化会损失掉原数据集的精度。另一方面这个数据集中的属性大部分都是考虑到隐私保护进行变换得到的，因此得到的规则可解释性也不强。

## 3. 不平衡数据探索

根据分类预测实验的结果，大多数算法对于欺诈样本的检测精度以及召回率非常低，这是由于该数据集的高度不平衡导致的。因此我们对不平衡数据进行了专门的探索研究，将完整的数据集按照0.8:0.2的比例划分为训练集和测试集，我们对于训练集分别使用不采样、欠采样和过采样的方式进行处理，其中过采样又分为随机过采样，合成少数类过采样(Smote)以及生成式对抗网络过采样（GAN)，分类预测算法统一采用逻辑斯蒂回归算法(Logistic Regression)。实验中所有采样的非欺诈类检测指标都等于1，因此不进行对比，评价指标统一采用欺诈类检测的F1分数，不同采样方法以及不同采样倍数的对比可视化结果如图所示。




根据实验结果，所有的采样方法都能够提高对欺诈样本的检测效果。欠采样方法在采样倍数很低时优于过采样方法，随着采样倍数的增加，过采样方法明显优于欠采样方法。随着欠采样倍数的增大使得数据集丢失了大量的信息，因此检测效果急剧下降；而随着过采样倍数的增大，使得算法过度的学习相同或相似的欺诈样本信息，导致算法的严重过拟合。总的来说，生成式对抗网络合成欺诈数据不仅能使得算法取得更好的效果，同时随着采样倍数的增加算法的过拟合并不严重，因此GAN能够稳定的解决数据集的不平衡问题。

## 总结

我们在项目中主要面临两个问题：1）由于隐私原因，特征V1-V28是经过PCA变换后的数据，我们无法获得元数据，所以很难进行频繁模式与关联规则挖掘的挖掘；2）实验所使用的数据集中正常交易与欺诈交易的比例接近99比1，也就是说该数据集高度不平衡，虽然在实验中容易获得很高的预测准确率，但其实是因为正常交易样本相对太多的影响，而欺诈交易的精确率和召回率可能很低。在小组成员的共同努力下，我们使用FP-growth算法进行关联分析，得知信用卡交易的时间与欺诈交易之间的关系并不是很强烈；对于欺诈交易的识别，我们采用了有监督方法和无监督方法，其中逻辑斯蒂回归和KNN可以更有效的识别欺诈交易。我们还更近一步的探索了如何处理不平衡数据集，在逻辑斯蒂回归上对多种采样方法（without sampling，under sampling，over sampling等）进行了测试。

总之，我们发现在数据挖掘任务中，绝大部分的工作是在对所需要挖掘的数据进行各种处理，其次根据已知的数据特点选择合适的算法/模型也非常重要。