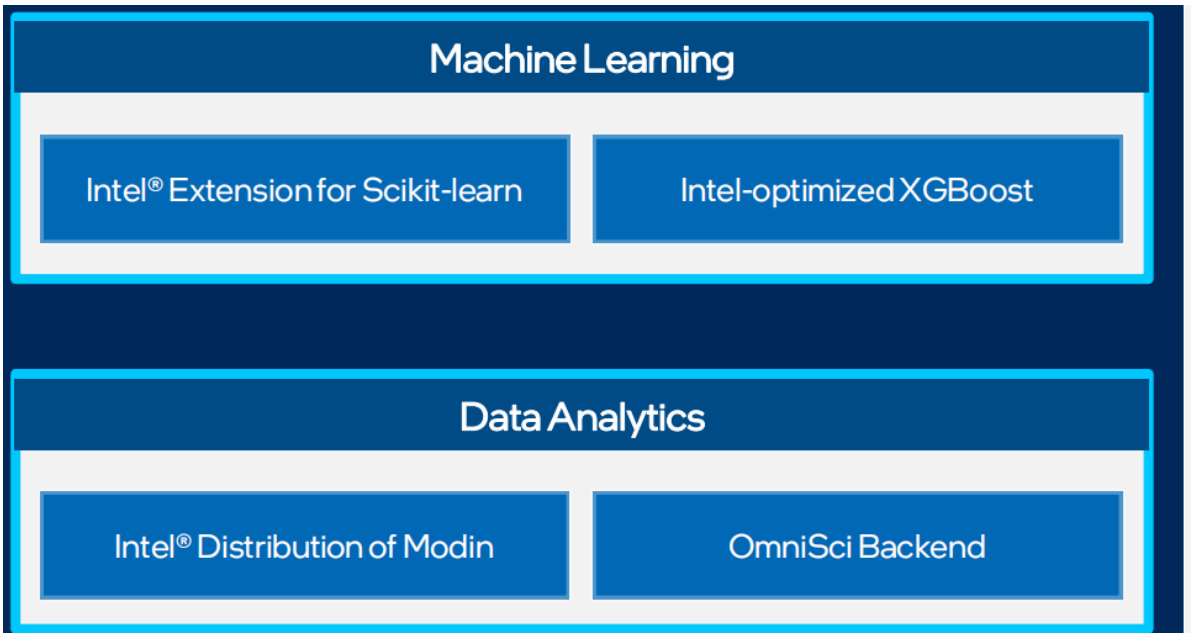
问题陈述

在日益数字化的环境中，信用卡欺诈的激增和复杂欺诈技术发展导致大量的金融损失。自动化欺诈检测变得越来越重要，目前已有许多工作利用传统的机器学习方法或是深度学习大大提升了检测准确率，但是如何**加快检测速度**，缩短响应时间这一挑战亟待解决。同时，由于欺诈事件发生的小概率性以及欺诈本身就受诸多方面影响，因此该问题数据集存在严重的**高度不平衡**以及**数据集维度较高**导致训练速度变慢等。

项目简介

项目以**探讨加快训练速度**为主要工作，使用Python语言，基于Intel的**modin模块 Intel® Extension for Scikit-learn**以及**Intel-optimized XGBoost**工具开发，分别对比了1）逻辑回归、随机森林和XGBoost三种方法在欺诈预测上的准确率；2）是否采用Intel工具开发的训练时间差距。

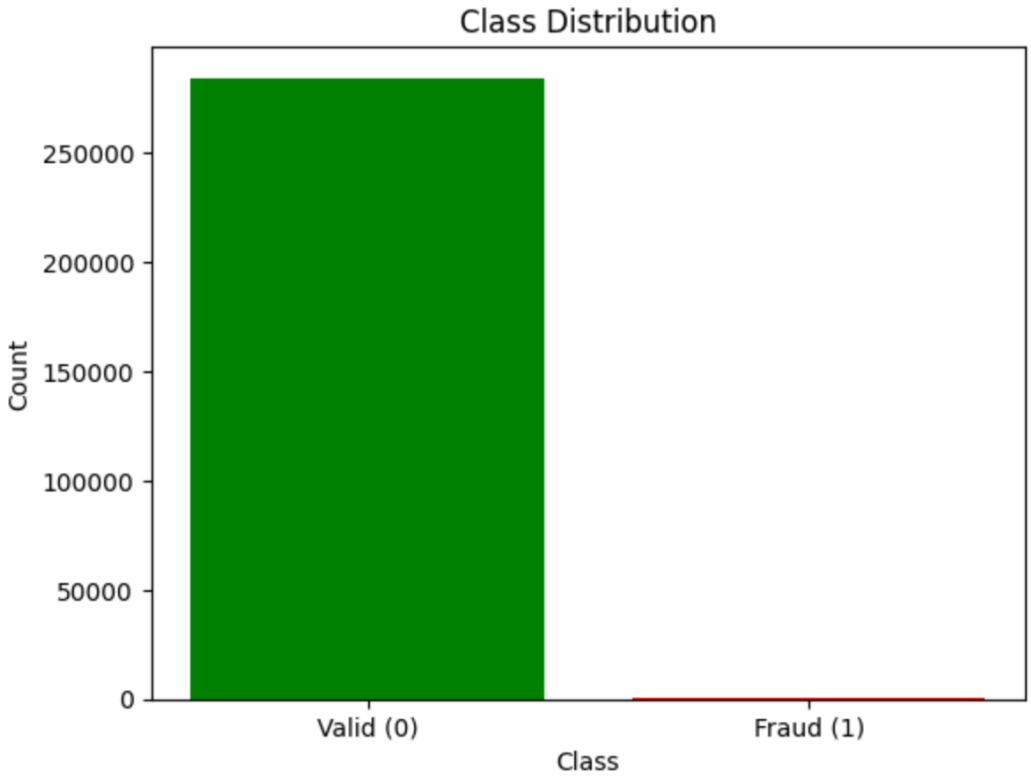


通过实验我们发现**随机森林**的预测效果最好，同时Intel加速工具在模型训练上能**大大减少运行时间。**

代码实现

数据预处理

首先我们先对数据检查是否有缺失和重复数据，实验发现缺失数据并不存在，但是存在1000多条重复数据，我们先将进行**去重**，同时通过打印数据分布可知数据集Class为1和Class为0存在**严重不平衡。**



在解决不平衡问题之前，我们还考虑到特征维度较高进行了相关性分析，结合相关系数大小以及实际情况考虑，排除了V13，V15, V26以及Time四个属性，减少特征维度以提高训练速率，同时对Amount进行了**标准化。**为了避免数据不平衡可能导致的模型在学习时花费更多的时间来适应多数类，而在少数类上的学习可能相对较少，从而对少数类的分类性能较差这种情况，我们对训练数据集（占70%）进行了**下采样**，最终数据分布如下：

	下采样之前	下采样之后
正常样本（0）所占整体比例	0.998	0.5
异常样本（1）所占整体比例	0.002	0.5
下采样策略总体样本数量	28w	20w

模型结果分析

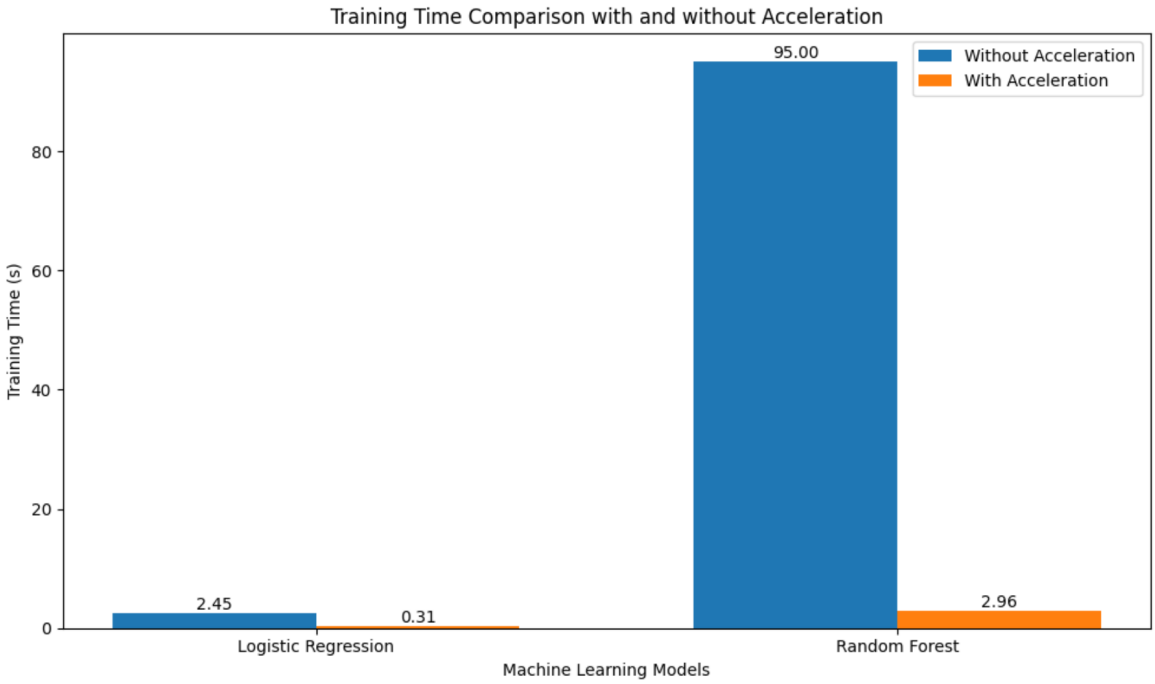
加速工具分析

我们通过如下代码示例分别比较Intel® Extension for Scikit-learn工具对模型训练的加速效果, 通过十次测验取平均时间得到结果如下图：

```
# 使用加速工具
from sklearnex import patch_sklearn
patch_sklearn()
start = timer()
rf_model.fit(X_resampled, y_resampled)
train_time = timer() - start
print(f"Intel® extension for Scikit-learn time: {train_time:.2f} s")

# 未使用加速工具
from sklearnex import unpatch_sklearn
unpatch_sklearn()
```

```
start = timer()
rf_model.fit(X_resampled, y_resampled)
train_time = timer() - start
print(f"Original Scikit-learn time: {train_time:.2f} s")
```

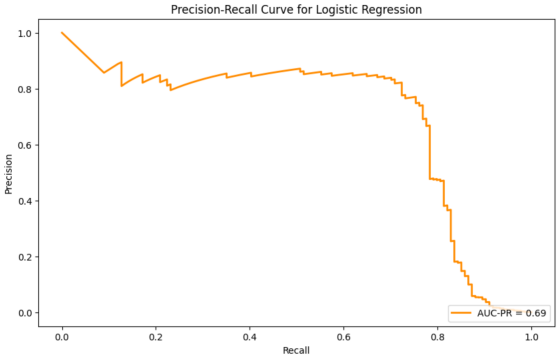


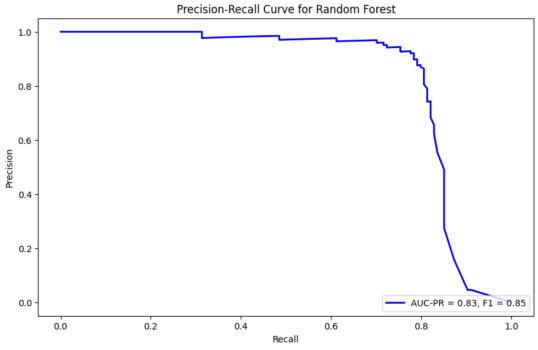
可以看到工具的使用大大加速了模型的训练，尤其是在随机森林方法中，我们在实验中发现使用加速工具可以将训练时间从90多秒减少至3秒左右！

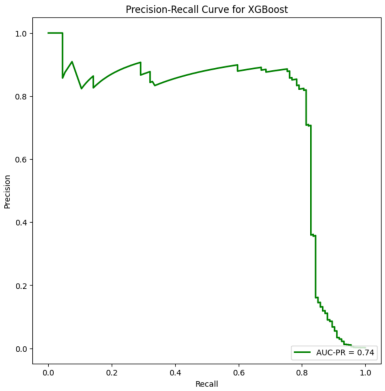
模型预测效果

AUC-PR（Area Under the Precision-Recall Curve）是 Precision-Recall 曲线下的面积，用于评估二元分类器的性能，实验通过计算AUPRC并绘制精确率-召回率曲线的面积（值越大越好）来分析模型预测好坏，此外我们还计算了F1分数，结果总结如下：

	AUC-PR	F1分数
逻辑回归	0.69	0.77
随机森林	0.83	0.85
XGBoost	0.74	0.81







团队收获

在此次项目中，团队成员对课堂上所讲的机器学习方法更加熟悉，同时更印象深刻的是采用Intel加速工具开发对模型训练的大大提高。在之后的项目开发中我们学习到了不仅要学会模型的选择以及调参，同时还需要利用好的工具加速项目进展。