第 3 章 医疗保险稽核

保险欺诈是在保险孕育之初就存在的重要问题，如今所有理赔案件中赔付率和赔付金额高居不下，并且有逐年上升的趋势，成为目前保险行业的一大隐患。

目前国内外的很多保险公司都实现了线上理赔，保单数据、理赔案件大量地在线上作业完成，这对于提高用户体验有很大的促进作用，同时也提升了保险公司的整体生产效率。但是在保险理赔风险把控上存在明显不足。如何有效提高风控能力是各大保险公司目前亟须解决的问题。

3.1 数据预处理

医疗保险稽核业务具有多年的业务历史，在保险稽核行业多年信息化建设中，保险公司积累了大量的保险数据信息，这些数据具有多维度、数值化数据多等特点。然而保险稽核数据存在大量冗杂属性、噪声、数据缺失和数据错误的情况。在开始数据挖掘工作之前，需要进行数据集成、数据清洗、数据转换、特征工程和数据平衡等方面的预处理。

3.1.1 特征选择

保险稽核的数据主要来源于案件信息 CLAIM、保单信息 POLICY、医疗账单信息 BILL、银行账户信息 ACCOUNT、理赔申请 APPLICATION 等复杂的数据关系表，其主要依靠报案号 REPORT_NO 联系在一起。对这些数据表格的整理得到的类如图 3.1 所示。

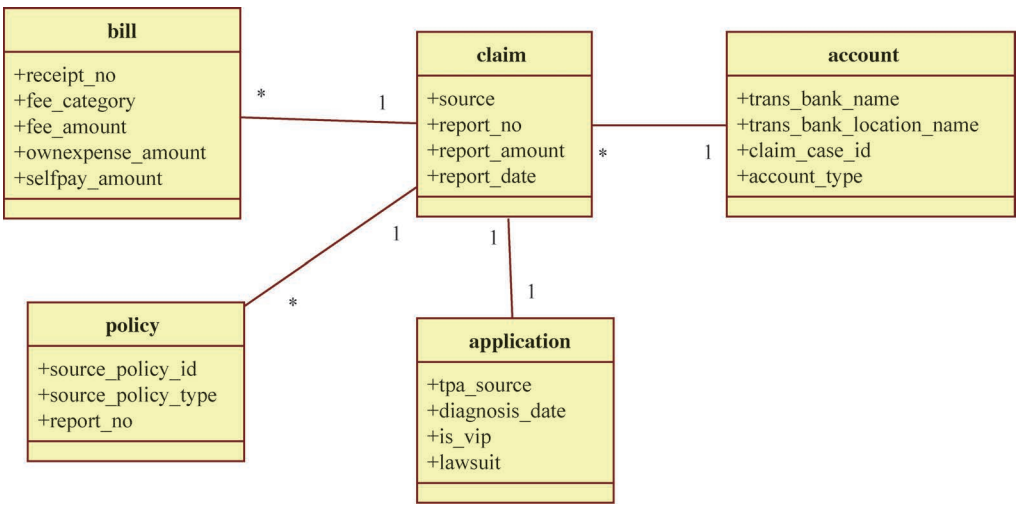

图 3.1 数据关系类图

选择 Pandas 库的 read_csv 读取数据并查看样本信息：

```
df = pd.read_csv('creditcard.csv')
df.info()
```

运行后结果如下：

```
<class'pandas.core.frame.DataFrame'>
RangeIndex:3512entries,0to3511
Datacolumns(total38columns):
#Column
----------
0id
1age
2customer_months
3policy_bind_date
4policy_state
5source
6loss_code
7area
…
30total_claim_amount
31injury_claim
32property_claim
33vehicle_claim
34auto_make
35auto_model
36auto_year
37fraud
dtypes:float64(1),int64(18),object(19)
memoryusage:897.9+KB
```

所有的字段并非都是需要的特征，需要分析这些字段对稽核案件的影响，去除对结果影响不大的字段，这有助于算法将训练的注意力集中在主要的特征上，提升模型的泛化能力。选择过程如下：

```
<class'pandas.core.frame.DataFrame'>
df = df[[
'age',
'customer_months',
'policy_bind_date',
'policy_state',
'phone',
'source',
'loss_code',
'area',
'insured_zip',
'insured_sex',
'insured_education_level',
'insured_occupation',
'insured_hobbies',
…
'incident_hour_of_the_day',
'number_of_vehicles_involved',
'property_damage',
'bodily_injuries',
```

```
    'witnesses',
    'police_report_available',
    'amount',
    'injury_claim',
    'property_claim',
    'vehicle_claim',
    'fraud']]
```

3.1.2 数据清洗

由于系统的不断升级换代,表结构会经常变化,这样会导致后续新增的字段在历史数据中缺失。前端限制需求在不断变化后,对数据的标准也会有影响。这样的数据如果直接应用到数据挖掘中,很容易造成模型结果的失真,所以在使用这些数据之前,需要进行数据清洗。

1. 缺失值处理

执行代码 df_null=df.isnull()后,可以得出每个特征的取值缺失比例如下:案件来源缺失笔数为1150,出险地区缺失笔数为1042,固定电话缺失笔数为32,出险经过缺失笔数为345,出险原因缺失笔数为560。对于这些缺失的数据,一般会采用插补和直接删除两种方案处理。

固定电话PHONE字段是系统的主要数据,在正常情况下是必填字段,不会出现这样的缺失值,在查找具体存库的时间和原因后,发现是系统Bug导致的,对于这样的数据,由于本身缺失的比例比较小,可以直接进行删除。代码如下:

```
df.dropna(self,axis = 0,how = 'any')
```

此外,还存在部分案件来源SOURCE、出险原因LOSS_CODE的缺失数据,由于案件来源和出险原因是通过下拉页面选择的,都属于分类变量,有选项范围和默认值,一般这种缺失值可以通过默认值和众数进行插补。可以通过使用KNN方法选择目标数据邻近的k个数据,对缺失目标的缺失值进行补充。此方法在处理缺失数据时的优点是快速、高效,在面对连续有较多缺失数据时会有降低准确性的风险。

```
def missingData(data):
    #基于业务填补
    data.SOURCE[data.SOURCE.isnull()] = data.SHRP[data.SOURCE.isnull()]
    data.LOSS[data.LOSS.isnull()] = data.SHRP[data.LOSS.isnull()]
    data = fitDGFNatureByCLAC(data)
    #KNN 填补
    return KNN(k = 3).fit_transform(data)
```

2. 错误数据处理

在分析数值型变量时,发现药品费用、诊疗费用、材料费用、手术费用、住院费用等与正常值有较大差距的值,为了不影响这些数据对模型的负面影响,需要先找出这些数据。对于这些数据进行标准化,去除绝对值对结果的影响,然后用箱线图的方式进行异常值的查找。其中,上边界以上、下边界以下的数据可以认为是异常数据。对异常的数据采用删除的方式。下面箱线图的代码展示了药品、诊疗、材料、手术、住院5个类别的数值分布,可见材料和手术中的数据有较多的异常值是偏大的。

```
path = 'claim.csv'
data = pd.read_csv(path, sep = ',')
drug = data['drug']
treatment = data['treatment']
material = data['material']
operation = data['operation']
in_hospital = data['in_hospital']
labels = {'药品','诊疗','材料','手术','住院'}
plt.boxplot([drug,treatment,material,operation,in_hospital],labels)
plt.show()
```

就诊金额数据的箱线图如图 3.2 所示。

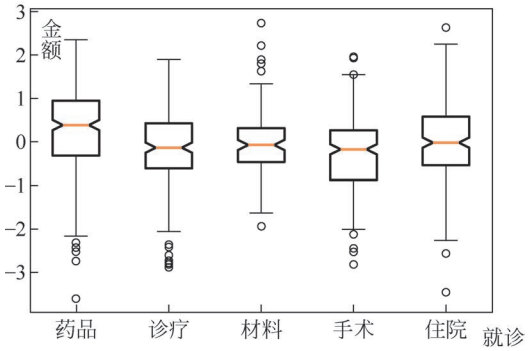

图 3.2　就诊金额数据的箱线图

3.1.3　数据离散化

稽核数据中有很多是连续的数据，对这些数据进行离散化，能够提高算法的效率。在分析药品金额时，金额的绝对值不利于理解，这里把药品金额按照数值的高低分为一、二、三、四、五 5 个档次，这样在分析每个理赔报案的药品金额时能够在全局上对该理赔报案人员进行把握。此外，把所有客户的报案次数进行离散化，可以分为一般用户、报案偏多用户和异常用户三种类型。应用 K-Means 算法对药品金额进行聚类离散化。

```
data = pd.read_csv(r'.\claim.csv',encoding = 'gbk')
x = data[['Shd','Amount']]
#聚类实现
kms = KMeans(n_clusters = 3)
#进行聚类,确定每一行的数据属于哪一类
y = kms.fit_predict(x)
```

3.1.4　特征值处理

在上述数据清洗后，还需要对数据进行降维，可以采用以下降维方法：因子分析、独立成分分析和主成分分析(PCA)。

通过对众多特征值的分析处理，删除对分类模型影响不大的特征：出险地点、案件来源、银行信息、缴费频率、医院等级等，选择主要的特征变量：出险原因、索赔金额、是否黑名单、药品/诊疗/材料/手术费用和总金额的比值。

3.1.5 数据平衡

fraud字段表示该样本是否为欺诈样本,使用以下代码对欺诈和非欺诈样本比例进行查看:

```
df['fraud'].value_counts()
```

结果如下:

```
99228
525
Name:fraud,dtype:int64
```

近10万条训练样本中,具有欺诈风险的数据,即数据集中类为1的数据有525条。欺诈风险案件和正常案件的比值是1∶189,这种数据分布是不平衡的。

对模型训练而言,如果欺诈的训练样本太少,容易导致分类模型缺少足够的学习样本,进而导致无法判断出少量样本的类。数据不平衡常用的方法是过采样和增加权值等方案。这里采用的是过采样,通过使用重复、自举或合成少数类过采样的SMOTE算法来生成新的欺诈风险案件的数据。

```
from imblearn.over_sampling import SMOTE
smo = SMOTE(random_state = 42)
df,label = smo.fit_sample(df,label)
```

通过SMOTE算法,系统将欺诈和非欺诈案件的数量比例设置为1∶1,以补充小类数据的不足。

3.1.6 样本权重系数设置

在保险欺诈场景下,将负样本预测为正样本比将正样本预测为负样本代价要大得多,因此在训练中可能需要对训练样本设置权重,让样本在训练时乘以一个系数。这里通过添加一个新定义的代价调整函数来实现。可以手动调参,例如将错误分类为正确的样本权重设置为7,正确分类为错误的样本权重设置为1。

```
#新定义的代价调整函数
def _beta(self,y,y_hat):
    res = []
    for i in zip(y,y_hat):
        if i[0] == i[1]:
            res.append(1)
        elif i[0] == 1 and i[1] == -1:
            res.append(7)
        elif i[0] == -1 and i[1] == 1:
            res.append(1)
        else:
        print(i[0],i[1])
        return np.array(res)
```

3.1.7 数据转换

自理、自费等项目的费用对于不同的疾病情况有不同的阈值参考,绝对值不能直接用于数据分析,也没有参考意义,因此利用它和发票总金额的比值作为特征值进行分析。

投保人与被保人的关系包括本人、配偶、子女、父母等多种情况。在分析欺诈风险的案件中，只要考虑本人和非本人就够了，具体的投保人和被保人的关系对结果的影响不大，所以投保人和被保人的关系主要分为本人和非本人。

3.2 医疗保险稽核建模和评估

在数据预处理后，可以进行模型的训练。本案例主要运用 AdaCostBoost 算法对稽核数据进行训练，并进行验证。

对比传统的 AdaBoost，其改进版 AdaCostBoost 算法更加注重将负样本错分类为正样本的情况。在 AdaBoost 训练的过程中，对于分类错误的样本，模型会为这个样本增加对应的权重，而不分正负样本，而改进的 AdaCostBoost 算法将对把负样本分类为正样本的情况放大，具体的做法是乘以 beta 系数，让模型更加关注将负样本错分类为正样本的情况。

sklearn 库中的 AdaBoostClassifier 用于分类，可以使用下面的语句实现模型的构建：

```
bdt = AdaBoostClassifier(algorithm = "SAMME",n_estimators = 200,learning_rate = 0.8)
bdt.fit(x,y)
```

通过 sklearn.metrics 中的 accuracy_score，recall_score，f1_score 可以得到 AdaBoost 模型的评估结果。

通过分析数据可知，AdaBoost 的总体准确率有 89%。然而在保险欺诈的场景下，该算法在错误分类时，对于负样本错判为正样本，或者正样本错判为负样本，由于代价不同，需要进行权值调整。在上一节准备了 beta 调整函数，通过新增的调整函数可以将错分样本按照代价大小再次做权值区分，_boost_real()中的权值需要乘上新增函数 self._beta(y,y_predict)。

```
sample_weight * = np.exp(estimator_weight * ((sample_weight > 0)|
(estimator_weight < 0)) * self._beta(y,y_predict))
```

通过调整 beta 系数，在精确性提升的同时召回率有了一定的下降，以 F1 作为一个整合的指标。对比可知，通过阈值的设置，改进后的算法的精确值和 F1 值更高了。

AdaBoostClassifier 算法中主要的参数有 n_estimators，即基分类器提升次数，默认是 50 次。n_estimators 过大或过小都会有不足：值过大，模型容易过拟合；值过小，模型容易欠拟合。

上面当 n_estimators=200，ACC 为 0.91，REC 为 0.86，F1 为 0.91 时，效果不是很好。通过增加弱分类的数量至 700，三个值分别是 0.92、0.84 和 0.91。由于在 n_estimators 取值 300 再新增弱分类器个数后，数据的精确度有所下降，故认为 n_estimators=300 时的效果最好。错误率随 n_estimators 的调参变化如图 3.3 所示。

通过上面的过程，发现 AdaCostBoost 这种自适应的模型与报案稽核模型判断的准确度是很契合的，能够满足稽核业务的预测要求，减少了大量人力的支持。

经过上述分析，说明在保险稽核业务汇总 AdaCostBoost 对于业务的分析是很合适的，能够有效地识别问题案件，并且获得比较高的准确性。

选用其他模型与 AdaCostBoost 进行对比。

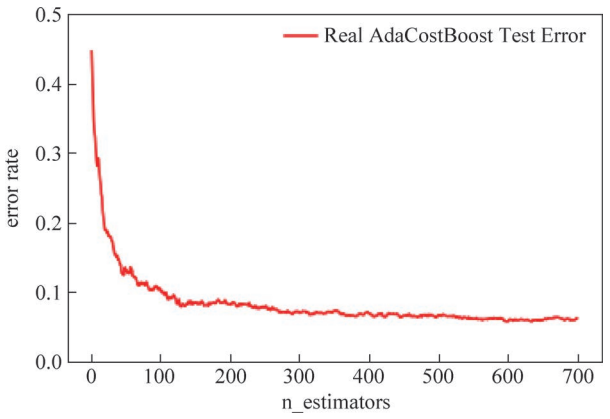

图 3.3　错误率调参曲线图

(1) SVM 模型：

```
from sklearn.svm import SVC
svm = SVC(C = 100,probability = True)
svm.fit(x,y)
svm_score = svm.score(valid_x,valid_y)
```

(2) 逻辑回归模型(LR)：

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
lr.fit(x,y)
lr_score = lr.score(valid_x,valid_y)
```

(3) 随机森林模型(RF)：

```
RF = RandomForestClassifier(max_features = None,bootstrap = False)
RF.fit(x,y)
RF_score = RF.score(valid_x,valid_y)
```

(4) KNN 模型(KNN)：

```
knn = KNeighborsClassifier(n_neighbors = 3)
knn.fit(x,y)
knn_score = knn.score(valid_x,valid_y)
```

(5) 决策树模型(DT)：

```
DT = DecisionTreeClassifier()
DT.fit(x,y)
DT_score = DT.score(valid_x,valid_y)
```

(6) 朴素贝叶斯模型(GS)：

```
GS = GaussianNB()
GS.fit(x,y)
GS_score = GS.score(valid_x,valid_y)
```

(7) 伯努利朴素贝叶斯模型(BNL)：

```
BNL = BernoulliNB()
BNL.fit(x,y)
BNL_score = BNL.score(valid_x,valid_y)
```

（8）XGBoost 模型：

```
import xgboost as xgb
xgb_train = xgb.DMatrix(x,y)
xgb_test = xgb.DMatrix(valid_x,valid_y)
xgb_model = xgb.train(dtrain = xgb_train,params = params)
xgb_predict = xgb_model.predict(xgb_train)
```

将多种模型与 AdaCostBoost 算法进行对比，比较的指标包括 ACC、REC、F1、AUC 等。其中部分模型的比较如表 3.1 所示。

表 3.1　算法比较

算　法	ACC	REC	F1	AUC
AdaCostBoost	0.92	0.84	0.91	0.97
SVM	0.88	0.79	0.88	0.96
LR	0.87	0.92	0.87	0.87
GS	0.86	0.72	0.84	0.85
XGBoost	0.85	0.77	0.87	0.88
BNL	0.89	0.84	0.87	0.90
RF	0.90	0.85	0.84	0.89

通过实验分析可知，在小样本的数据训练中，SVM 的效率更高，但是准确率比较低，毕竟 AdaCostBoost 算法是一种自适应的算法，通过大量的数据和多种弱分类不断自适应后，AdaCostBoost 的效果要更好。而在保险稽核这种核心业务中，精准性是优先考虑的因素，着重关注对于欺诈案件的稽核，实际业务中宁可错分正常案件为欺诈案件，也不愿漏过真正有欺诈风险的报案。几种算法的性能比较如图 3.4 所示。

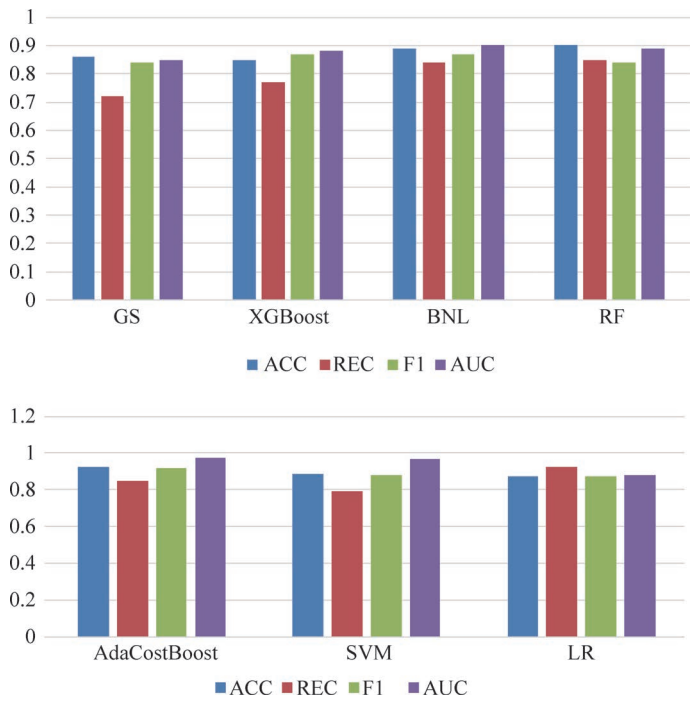

图 3.4　不同模型保险稽核预测比较

3.3 结果分析

通过分析实验结果可知,对于准确率指标,AdaCostBoost 最高有 92%,比 SVM 和 LR 都高,比较符合模型要求,由于提高了准确率,因此召回率有所降低。以 AUC 值作为最终的判别标准来看,AdaCostBoost 达到了最高的 97%,AdaBoost、AdaCostBoost(靠上的曲线)的 ROC 曲线图如图 3.5 所示。在模型评估时,准确率不是唯一的指标,需要结合召回率、AUC 值和 F1 值综合考虑。在召回率可以接受的情况下,准确率最高的 AdaCostBoost 算法能有效地降低欺诈风险案件误判的概率。

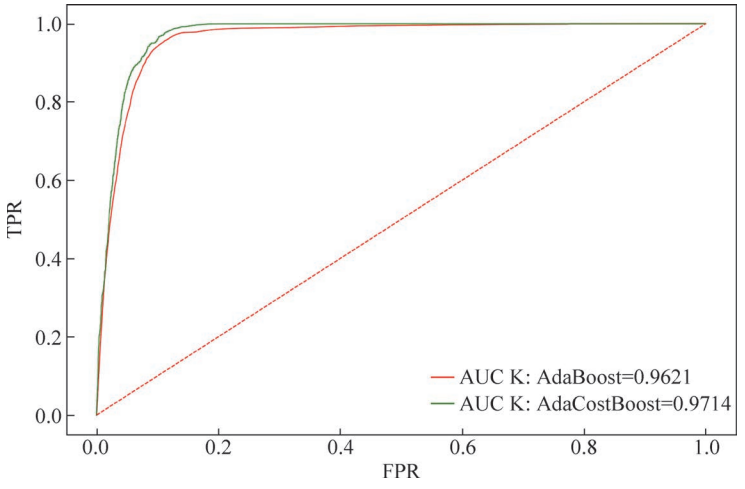

图 3.5 不同模型的保险稽核算法 ROC 曲线

思考题

1. 讨论如何对连续变量进行离散化。
2. 讨论如何处理数据的噪声。
3. 举例说明 SMOTE 算法的作用。
4. 如何比较不同分类算法性能的优劣。
5. 与 AdaBoost 算法相比,AdaCostBoost 算法做了哪些改进?