基于模型构建与特征选择的肝病分类研究

Abstract

本研究旨在肝癌、肝炎、肝硬化患者的肝功能检查指标对肝病患者的分类效果，通过分析临床特征并采用多种特征选择方法，研究旨在识别最具分类效果的关键特征。我们对数据进行了预处理，包括缺失值处理、特征缩放和特征选择，以确保模型的高效表现。研究中以互信息、递归消除特征和随机森林分类模型评估了各个特征对于诊断肝部疾病的影响。本论文展示了机器学习技术在医学领域中的应用，并强调了特征工程在预测建模中的重要性。

1. Introduction

特征选择，是为了降低特征空间维度，选择出一些最有效特征的过程，是模式识别的关键步骤之一。[1]对于一个模式识别系统，一个好的学习样本是训练分类器的关键，样本中是否含有不相干或冗余信息是分类器训练的关键。

肝病，包括肝炎、肝硬化和肝癌，是全球范围内的重大健康问题，对患者的预后和治疗有着重要影响。早期准确的诊断对提高患者的治疗效果至关重要，但传统的诊断方法往往既费时又昂贵。随着机器学习技术的进步，基于临床数据自动化分类肝病的研究逐渐成为可能。本文旨在探索多种机器学习算法在肝炎、肝硬化和肝癌分类中的应用，重点分析哪些临床特征对于准确预测肝病最为关键。通过一系列的数据预处理步骤和模型评估，分析讨论了三种主流特征选择方法对肝癌患者的肝功能指标特征选择的效果及比较。

2. Data Processing

利用 pandas 库将 data.csv 进行导入，再根据不同种类的疾病对样本分别使用线性插值进行缺失值处理，最后将样本整合并导出特征值矩阵以及目标矩阵。

```
import pandas as pd

from sklearn.model_selection import train_test_split

import numpy as np

from sklearn.ensemble import RandomForestClassifier

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import accuracy_score
```

```
from sklearn.feature_selection import RFE

df = pd.read_csv('data.csv')

df = df.applymap(lambda x: np.nan if x==0 else x)

grouped = df.groupby('诊断名称')

sub_dfs = {name: group for name, group in grouped}

sub_dfs['肝硬化'].interpolate(method='linear', inplace=True)

sub_dfs['肝炎'].interpolate(method='linear', inplace=True)

sub_dfs['肝癌'].interpolate(method='linear', inplace=True)

df = pd.concat([sub_dfs['肝硬化'], sub_dfs['肝癌'], sub_dfs['肝炎']])

df = df.dropna()

df_dict = {'data': df.drop(['ID','诊断名称'], axis=1), 'target': df['诊断名称']}

X = df_dict['data'].to_numpy()

y = df_dict['target'].to_numpy()

scaler = StandardScaler()

X_scaled = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.1, random_state=42)
```

3. Methods

3.1 过滤式

信息增益定义为先验不确定性与期望的后验不确定性之间的差异,它能有效地选出关键特征,剔除无关特征。互信息描述的是两个随机变量之间相互依存关系的强弱。[2]

```
from sklearn.feature_selection import mutual_info_classif

mi = mutual_info_classif(X, y)

# 获取互信息值的排序索引

sorted_indices = np.argsort(-mi)

# 获取排序后的互信息值

sorted_mi = mi[sorted_indices]

n_repeats = 10

feature_names = df_dict['data'].columns.tolist()

# 初始化一个列表，用于存储每个特征的累计互信息值

cumulative_mi = np.zeros(len(feature_names))

# 多次计算互信息并累加

for _ in range(n_repeats):

    mi = mutual_info_classif(X, y)

    cumulative_mi += mi

# 获取排序后的互信息值索引

sorted_indices = np.argsort(-cumulative_mi)

# 打印排序后的互信息值及其原始位置

i = 0

print("Cumulative Mutual Information for each feature:")

for idx in sorted_indices:

    print(f"Feature {feature_names[idx]}: Cumulative Mutual Information = {cumulative_mi[idx]}")
```

```
    i = i + 1
    if i == 9:
        break
```

3.2 包裹式

把特征选择与分类器设计集成在一起，利用分类器进行特征选择。包裹式方法直接把最终将要使用的学习器的性能作为特征子集的评价准则。这里选用随机森林分类函数作为基函数，对特征进行递归消除进行筛选。

```
model = RandomForestClassifier()
model.fit(X_train, y_train)
importances = model.feature_importances_
feature_importances = pd.DataFrame({'feature': df.drop(['ID', '诊断名称'], axis=1).columns, 'importance': importances})
feature_importances = feature_importances.sort_values(by='importance', ascending=False)
N = 40
num_repeat = 3
acc_list =[[0]*len(feature_importances) for i in range(num_repeat)]
print('***************')
for j in range(num_repeat):
    print('-------------------------------------------')
    for i in range(40):
        selected_features = feature_importances.head(N-i)['feature']
        X_train_selected = X_train[:, selected_features.index]
        X_test_selected = X_test[:, selected_features.index]
        model_selected = RandomForestClassifier(random_state=42)
        model_selected.fit(X_train_selected, y_train)
        y_pred_selected = model_selected.predict(X_test_selected)
        acc_selected = accuracy_score(y_test, y_pred_selected)
        print(f'{N-i} 使用选定特征的准确率：{acc_selected}' )
        acc_list[j].append(acc_selected)
    print('-------------------------------------------')
cumulative_sum_columns = np.cumsum(X, axis=0).sum(axis=0)
acc_max = max(cumulative_sum_columns)
print(acc_max)
max_index = cumulative_sum_columns.index(acc_max)
print(max_index+1)
print(feature_importances[:max_index+1])
```

3.3 嵌入式

嵌入式特征选择是将特征选择过程与学习器训练过程融为一体，两者在同一个优化过程中完成，在学习器训练过程中自动地进行特征选择。这里将 RFC 与随机森林分类器进行结合以达到筛选特征的目的。

```
from sklearn.feature_selection import SelectFromModel
from sklearn.ensemble import RandomForestClassifier as RFC
RFC_ = RFC(n_estimators=10, random_state=0)
X_embedded = SelectFromModel(RFC_, threshold=0.005).fit_transform(X, y)
selected_features_mask = selector.get_support()
selected_feature_names = [feature_names[i] for i in range(len(feature_names)) if selected_features_mask[i]]
print("Selected features:")
print(selected_feature_names)
print("Number of selected features:", sum(selected_features_mask))
```

4. Results

综合以上三种方法，得出胆碱酯酶(ChE)-静脉血、血小板计数(PLT#)-静脉血、血小板比容-静脉血是诊断肝病的主要有效特征。

过滤式结果	包裹式结果	嵌入式结果
胆碱酯酶(ChE)-静脉血 0.069706	胆碱酯酶(ChE)-静脉血 1.105623789065392	总胆红素(TBIL)-静脉血
血小板计数(PLT#)-静脉血 0.057907	血小板计数(PLT#)-静脉血 1.0248667555525202	胆碱酯酶(ChE)-静脉血
血小板比容-静脉血 0.046705	血小板比容-静脉血 0.9957505652859344	红细胞分布广度-SD-静脉血
天门冬氨酸氨基转移酶/丙氨酸氨基转移酶比值(AST/ALT)-静脉血 0.040405	红细胞分布广度-SD-静脉血 0.7317568241251706	血小板比容-静脉血
总胆红素(TBIL)-静脉血 0.037988	间接胆红素(IBIL)-静脉血 0.6904131351449017	血小板计数(PLT#)-静脉血

表 1. 三种模型排列的前五个有效特征

5. Reference

[1]姚 旭,王晓丹,张玉玺,权文.特征选择方法综述，控制与决策，第 27 卷第 2 期
[2]姚 旭,王晓丹,张玉玺,权文.特征选择方法综述，控制与决策，第 27 卷第 2 期