人工智能与机器学习报告

1. 双向障碍检测

3220101111 洪晨辉

1. 实验介绍

1.1 实验背景

双相障碍属于心境障碍的一种疾病，表示既有躁狂发作又有抑郁发作的一类疾病。当前研究发现，在双相障碍发生过程中遗传因素、环境或应激因素之间的交互作用、以及交互作用的出现时间点等都产生重要的影响。

双相障碍检测，即通过医学检测数据预测病人是否双相障碍，或双相障碍治疗是否有效。医学数据包括医学影像数据与肠道数据。由于缺少医学样本且特征过多，因此选取合适的特征对双模态特征进行整合并训练合适的分类器进行模型预测具有较强的现实需求与医学意义。本实验力图完成少样本、多特征下的监督学习。

1.2 实验要求

a) 实现双模态特征选择与提取整合。

b) 选择并训练机器学习模型进行准确分类。

c) 分析不同超参数以及特征选择方法对模型的结果影响。

1.3 实验环境

Python3. Numpy 库进行相关数值运算，sklearn 库进行特征选择和训练机器学习模型等。

2. 实验操作

2.1 导入数据

```
Feature1 表的 shape: (39, 6670)
Feature2 表的 shape: (39, 377)
label    表的 shape: (39, 1)
```

	0	1	2	3	4	5	6	7	8	9	...	6660	6661	6662	6663
0	0.816394	0.313184	0.437542	0.421138	0.54941	0.740194	-0.097087	0.005081	0.009196	0.105606	...	0.548366	0.122165	0.289676	0.21173

1 rows × 6670 columns

	0	1	2	3	4	5	6	7	8	9	...	367	368	369	370	371	372	373	374	375	376
0	100.0	0.0	0.0	0.38706	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.07677	0.0	0.0	21.16578	0	0.0

1 rows × 377 columns

	0
0	1

可以看到，医疗数据中的样本和特征数量存在着极大的不平衡。其中医疗影像数据共 6670 维，肠道数据共 377 维，而样本仅有 39 个，其中正样本标签为 1 ，负样本标签为 -1 。因此，特征的筛选和组合以及机器学习模型的选择优化对提高模型的性能极其重要。

2.2 准备数据

数据预处理是一种数据挖掘技术，它是指把原始数据转换成可以理解的格式。在这个过程中一般有数据清洗、数据变换、数据组织、数据降维和格式化等操作。对于本数据集，没有无效或丢失的条目；然而需要我们进行特征的筛选和整合。这些预处理可以极大地帮助我们提升机器学习算法模型的性能和预测能力。本例中主要是归一化：

```python
from sklearn.preprocessing import MinMaxScaler


def processing_data(data_path):
    """
    数据处理
    :param data_path: 数据集路径
    :return: feature1,feature2,label: 处理后的特征数据、标签数据
    """

    #导入医疗数据
    data_xls = pd.ExcelFile(data_path)
    data={}

    #查看数据名称与大小
    for name in data_xls.sheet_names:
            df = data_xls.parse(sheet_name=name,header=None)
            data[name] = df

    #获取 特征1 特征2 类标
    feature1_raw = data['Feature1']
    feature2_raw = data['Feature2']
    label = data['label']


    # 初始化一个 scaler，并将它施加到特征上
    scaler = MinMaxScaler()
    feature1 = pd.DataFrame(scaler.fit_transform(feature1_raw))
    feature2 = pd.DataFrame(scaler.fit_transform(feature2_raw))

    return feature1,feature2,label
```

```python
#数据路径
data_path = "DataSet.xlsx"

#获取处理后的特征数据和类标数据
feature1,feature2,label = processing_data(data_path)

# 显示一个经过缩放的样例记录
display(feature1.head(n = 1))
display(feature2.head(n = 1))
```

	0	1	2	3	4	5	6	7	8	9	...	6660	6661	6662
0	0.811348	0.0	0.414645	0.201443	0.35027	0.662461	0.289633	0.372597	0.316459	0.49556	...	0.855183	0.670516	0.813051

1 rows × 6670 columns

	0	1	2	3	4	5	6	7	8	9	...	367	368	369	370	371	372	373	374	375	376
0	0.998685	0.0	0.0	0.12642	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.019504	0.0	0.0	0.771442	0.0	0.0

1 rows × 377 columns

2.3 特征选择

使用监督学习算法的一个重要的任务是决定哪些数据特征能够提供最强的预测能力。专注于少量的有效特征和标签之间的关系，我们能够更加简单

具体地理解标签与特征之间的关系，这在很多情况下都是十分有用的。

可以看到：医疗数据中的样本和特征数量存在着极大的不平衡，其中医疗影像数据共 6670 维，肠道数据共 377 维，而样本仅有 39 个。因此，为了训练预测模型，特征的筛选和组合以及机器学习模型的选择优化极其重要。本报告采用简单的皮尔逊相关系数法进行双模态选择融合，两者共计选择十个特征。

```
def feature_select(feature1, feature2, label):
    """
    特征选择
    :param  feature1,feature2,label: 数据处理后的输入特征数据、标签数据
    :return: new_features,label:特征选择后的特征数据、标签数据
    """

    # 统计特征值和label的皮尔孙相关系数  对两类特征分别进行排序筛选特征
    select_feature_number = 5
    select_feature1 = SelectKBest(lambda X, Y: tuple(map(tuple,np.array(list(map(lambda x:pearsonr(x, Y), X.T))).T)),
                                  k=select_feature_number
                                  ).fit(feature1, np.array(label).flatten()).get_support(indices=True)

    select_feature2 = SelectKBest(lambda X, Y: tuple(map(tuple,np.array(list(map(lambda x:pearsonr(x, Y), X.T))).T)),
                                  k=select_feature_number
                                  ).fit(feature2, np.array(label).flatten()).get_support(indices=True)

    # 查看排序后特征
    print("select feature1 name:", select_feature1)
    print("select feature2 name:", select_feature2)

    # 双模态特征选择并融合
    new_features = pd.concat([feature1[feature1.columns.values[select_feature1]],
                              feature2[feature2.columns.values[select_feature2]]],axis=1)
    print("new_features shape:",new_features.shape)
    # 返回筛选后的数据
    return new_features,label
```

```
#查看特征选择结果
new_features,label=feature_select(feature1, feature2, label)
print("特征 shape: ", new_features.shape)
```

最终选出如下十个特征：

```
select feature1 name: [1242 2060 2064 3290 3912]
select feature2 name: [ 30  56  77 134 247]
new_features shape: (39, 10)
特征 shape:  (39, 10)
```

2.4 混洗、切分数据

现在特征选择已经完成并得到了新的特征数据。那么下面将数据（包括特征和它们的标签）整合并切分成训练集和测试集。其中 80% 的数据将用于训练和 20% 的数据用于测试。然后再进一步把训练数据分为训练集和验证集，用来选择和优化模型。

```
from sklearn.model_selection import train_test_split


def data_split(features, label):
    """
    数据切分
    :param features: 特征选择后的输入特征数据
    :param label: 标签数据
    :return: X_train:数据切分后的训练数据
             X_val:数据切分后的验证数据
             X_test:数据切分后的测试数据
             y_train:数据切分后的训练数据标签
             y_val:数据切分后的验证数据标签
             y_test:数据切分后的测试数据标签
    """
    # 将 features 和 label 数据切分成训练集和测试集
    X_train, X_test, y_train, y_test = train_test_split(features, label, test_size=0.2, random_state=2020, stratify=label)

    # 将 X_train 和 y_train 进一步切分为训练集和验证集
    X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2, random_state=2020, stratify=y_train)

    return X_train, X_val, X_test, y_train, y_val, y_test
```

```
# 进行数据切分
X_train, X_val, X_test, y_train, y_val, y_test = data_split(new_features, label)

# 显示切分的结果
print("Training set has {} samples.".format(X_train.shape[0]))
print("Validation set has {} samples.".format(X_val.shape[0]))
print("Testing set has {} samples.".format(X_test.shape[0]))
```

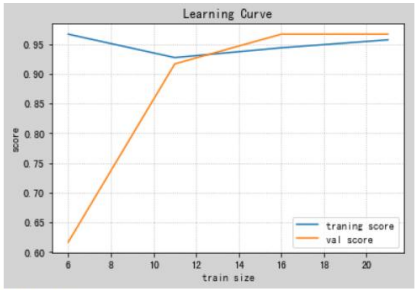
```
Training set has 24 samples.
Validation set has 7 samples.
Testing set has 8 samples.
```

共 39 个样本，选择 24/7/8 分的方式，合理。

2.5 监督模型选择

监督学习，也称为监督机式器学习，是机器学习和人工智能的一个子类别。它的定义是使用标记的数据集来训练算法，以准确分类数据或预测结果，符合本任务要求。本报告使用支持向量机，导入 scikit-learn 库完成模型训练。

```
from sklearn.model_selection import GridSearchCV,KFold
from sklearn.metrics import make_scorer

def search_model(X_train, y_train,X_val,y_val, model_save_path):
    """
    创建、训练、优化和保存深度学习模型
    :param X_train, y_train: 训练集数据
    :param X_val,y_val: 验证集数据
    :param save_model_path: 保存模型的路径和名称
    """

    #创建监督学习模型 以支持向量机为例
    clf = svm.SVC()

    # 创建调节的参数列表
    parameters = {
        'C': [0.1, 1, 2, 3, 5],
        'kernel': ['linear', 'rbf', 'poly'],
        'gamma': ['scale', 'auto'],
        'degree': [2, 3, 4, 5],
        'probability': [True]
    }

    # 创建一个fbeta_score打分对象 以F-score为例
    scorer = make_scorer(fbeta_score, beta=1)

    # 在分类器上使用网格搜索，使用'scorer'作为评价函数
    kfold = KFold(n_splits=10) #切割成十份

    # 同时传入交叉验证函数
    grid_obj = GridSearchCV(clf, parameters, scorer, cv=kfold)

    #绘制学习曲线
    plot_learning_curve(clf, X_train, y_train, cv=kfold, n_jobs=4)

    # 用训练数据拟合网格搜索对象并找到最佳参数
    grid_obj.fit(X_train, y_train)

    # 得到estimator并保存
    best_clf = grid_obj.best_estimator_
    joblib.dump(best_clf, model_save_path)

    # 使用没有调优的模型做预测
    predictions = (clf.fit(X_train, y_train)).predict(X_val)
    best_predictions = best_clf.predict(X_val)
```




```
best_clf
------
SVC(C=1, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=2, gamma='scale', kernel='linear',
    max_iter=-1, probability=True, random_state=None, shrinking=True, tol=0.001,
    verbose=False)

Unoptimized model
------
Accuracy score on validation data: 0.7143
Recall score on validation data: 1.0000
F-score on validation data: 0.7500

Optimized Model
------
Final accuracy score on the validation data: 0.7143
Recall score on validation data: 1.0000
Final F-score on the validation data: 0.7500
```

得到模型。值得注意的是，优化后的模型并没

有表现更好。同时，注意到模型因为训练数据集太少，出现了较为严重的过拟合现象（训练分数近于 1 的情况下，验证集仅有 0.75 左右）。

3. 结果分析

提交测试，得到：

结果尚可。这与该模型在训练过程中的良好表现不无关系。

4. 总结与讨论

本次报告探索并实践了一系列数据处理函数，对双向障碍患者的数据库进行了全面的分析和预测，仅使用了 10 个特征，取得显著成效。本实验力图教授数据降维的有效技巧和主成分提取的实用方法，将极大地帮助未来的研究和工作中更高效地处理和分析数据。