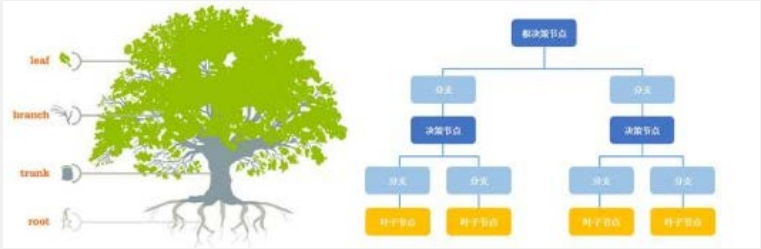


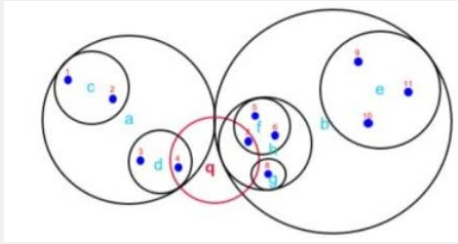
地震预测算法模型汇报

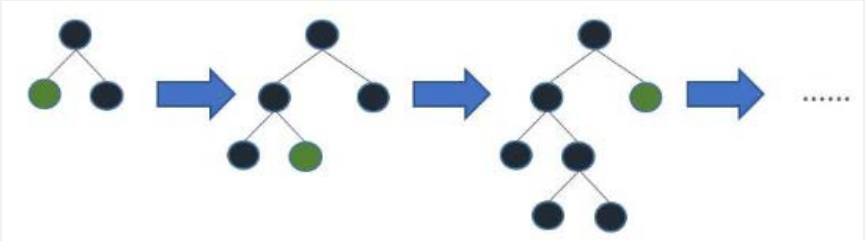
小组：时光倒流

目录
contents

控制目标流程







数据处理 → 特征提取 → 模型分析 → 应用可视化

模型分析 → 对比分析

对比分析 → 模型再训练 → 特征提取

第一章
Part 01

数据处理

台站数据处理

台站数据通过运算官方提供筛选四川台站

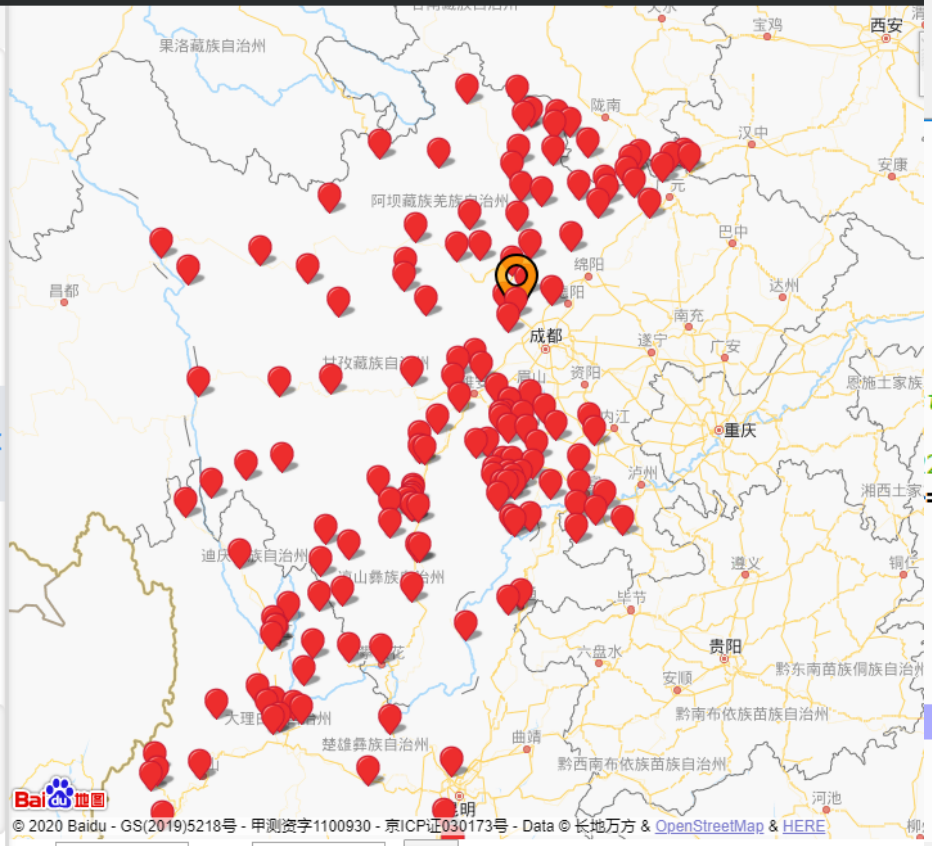
源代码 运行 刷新 帮助

```
  })
});
  map.addOverlay(vectorMarker);
  map.setViewport({center:new BMap.Point(103.65,30.98),zoom:14})
  function hide(){
  vectorMarker.hide();
}
function show(){
  vectorMarker.show();
}
    function openInfo(content,e){
        var p = e.target;
        var point = new BMap.Point(p.getPosition().lng,p.getPosition().lat);
        var infoWindow = new BMap.InfoWindow(content,opts);
// 创建信息窗口对象

map.openInfoWindow(infoWindow,point); //开启信息窗口
    }
  function theLocation(){

if(document.getElementById("longitude").value != "" && document.getElementById("latitude").value != ""){

map.clearOverlays();
            var new_point = new
```




```
t.xlsx')

2)]
= 98)]
```

地震数据处理

全球发生地震数据：通过运算，根据官方给的范围，筛选出四川地区的地震数据

```
import pandas as pd
import warnings
import numpy as np
import ssl
warnings.filterwarnings('ignore')
pd.set_option('display.max_columns', 100)

df = pd.read_csv('D://di//A榜训练数据和测试数据//train/eq_list_train.csv')
df['Magnitude'] = df['Magnitude'].apply(np.ceil)
# print(df['Magnitude'].unique())
df = df[(df['Latitude'] <= 34) & (df['Latitude'] >= 22)]
df = df[(df['Longitude'] <= 107) & (df['Longitude'] >= 98)]
list_area = [1 for i in range(0,len(df))]
df.loc[:,'area'] = list_area
df['area'] = df['area'].apply(int)
print(df)
# plt.scatter(df['Longitude'], df['Latitude'])
# plt.show()
df.to_pickle('D://di//final6//label.pkl')
```

训练数据处理

数值范围限制

合理使用低精度类型替换高精度类型

不损伤原数据精度

降低整体的数据空间占用

```
def reduce_mem(df):
    start_mem = df.memory_usage().sum() / 1024 ** 2#内存优化
    for col in df.columns:
        col_type = df[col].dtypes
        if col_type != object:
            c_min = df[col].min()
            c_max = df[col].max()
            if str(col_type)[:3] == 'int':
                if c_min > np.iinfo(np.int8).min and c_max < np.iinfo(np.int8).max
                    df[col] = df[col].astype(np.int8)
                elif c_min > np.iinfo(np.int16).min and c_max < np.iinfo(np.int16)
                    df[col] = df[col].astype(np.int16)
                elif c_min > np.iinfo(np.int32).min and c_max < np.iinfo(np.int32)
                    df[col] = df[col].astype(np.int32)
                elif c_min > np.iinfo(np.int64).min and c_max < np.iinfo(np.int64)
                    df[col] = df[col].astype(np.int64)
            else:
                if c_min > np.finfo(np.float16).min and c_max < np.finfo(np.float16
                    df[col] = df[col].astype(np.float16)
                elif c_min > np.finfo(np.float32).min and c_max < np.finfo(np.float
                    df[col] = df[col].astype(np.float32)
                else:
                    df[col] = df[col].astype(np.float64)
    end_mem = df.memory_usage().sum() / 1024 ** 2
    print('{:.2f} Mb, {:.2f} Mb ({:.2f} %)'.format(start_mem, end_mem, 100 * (start
    gc.collect()
    return df#通过数值范围限制，合理使用低精度类型替换高精度类型，使得即不损伤原数据精度，又可
```

训练数据处理

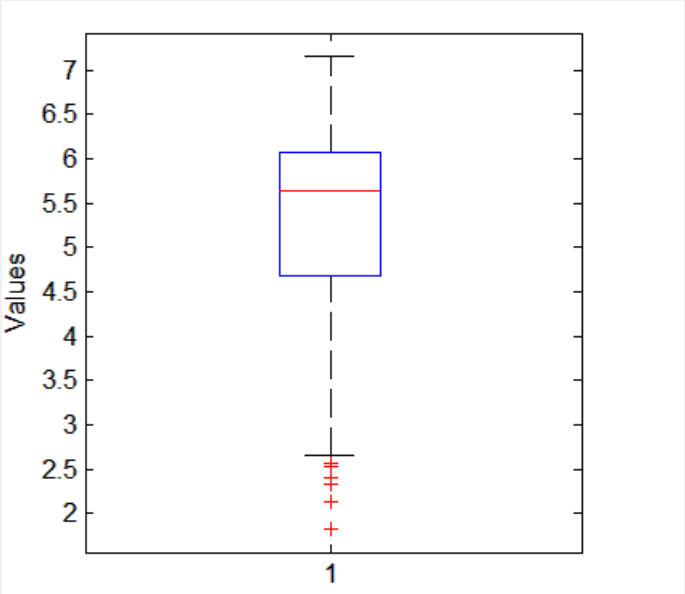
用当前数据减均值的方法，降低数据量

整合所有电磁地声数据

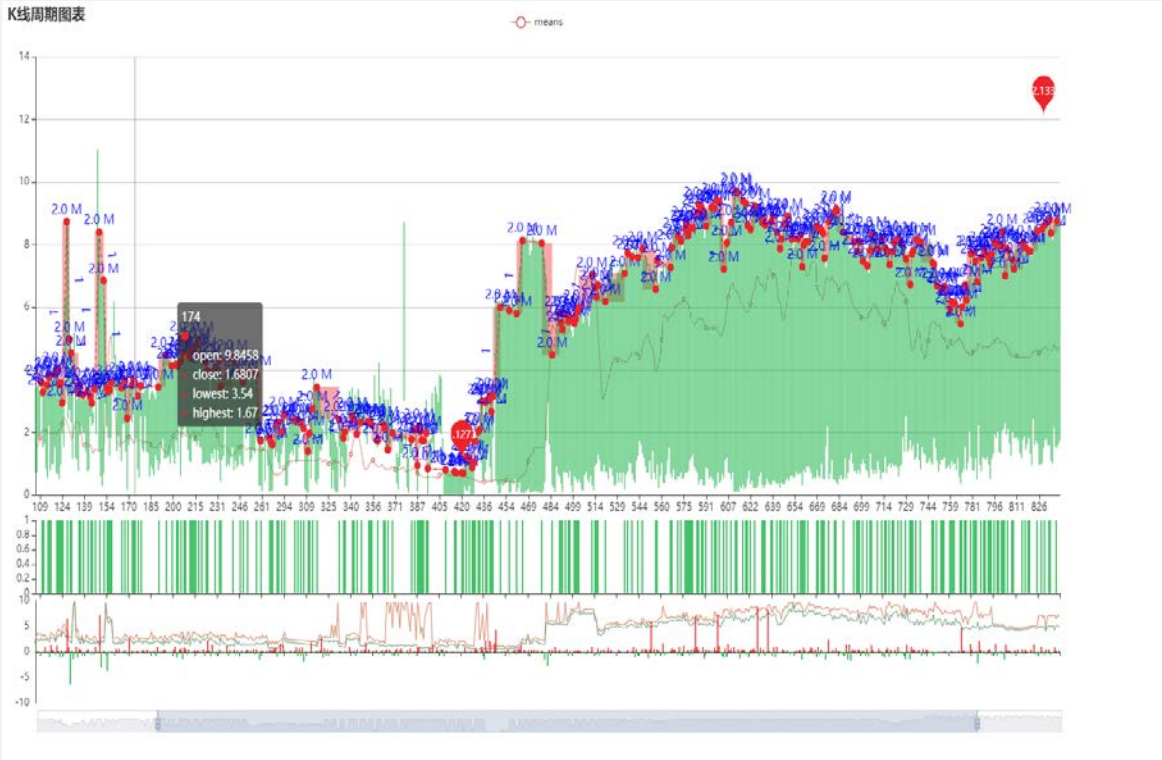
```
magn = []
sound = []
for file in tqdm(train_files):
    i = 0
    for f in os.listdir(f'{train_path}/{file}'):
        if i == 0:
            df = pd.read_csv(f'{train_path}/{file}/{f}')
            df['average'] = df['average'] - df['average'].mean()
            magn.append(df)
            i += 1
        else:
            df = pd.read_csv(f'{train_path}/{file}/{f}')
            df['average'] = df['average'] - df['average'].mean()
            sound.append(df)

df_magn = pd.concat(magn)
df_sound = pd.concat(sound)
df_magn = reduce_mem(df_magn)
df_sound = reduce_mem(df_sound)
df_magn.to_pickle('D://di//A榜训练数据和测试数据//train_magn.pkl')
df_sound.to_pickle('D://di//A榜训练数据和测试数据//train_sound.pkl')
print(df_magn.head())
```

每个台站异常数据处理



箱线图（离群值）



台站的历史观测图（异常点和缺失值）

第二章

Part 02

特征提取

数据整合

根据stationID，电磁地声的数据与台站数据用merge方法合并。

```
def feature(flag='train'):
    print(flag)
    if flag == 'train':
        df = pd.read_pickle('D://di//A榜训练数据和测试数据//train_sound.pkl')
        df1 = pd.read_pickle('D://di//A榜训练数据和测试数据//train_magn.pkl')
    else:
        df = pd.read_pickle('D://di//final6//test_a_sound.pkl')
        df1 = pd.read_pickle('D://di//final6//test_a_magn.pkl')

    loc_df = pd.read_pickle('D://di//final6//Stationid_list.pkl')
    df = df.merge(loc_df, on='StationID', how='left')
    # loc_df1 = pd.read_pickle('D://di//final_Data//Stationid_list1.pkl')
    df1 = df1.merge(loc_df,on='StationID', how='left')
    print(df1)
```

```
test
        StationID   average                 Time Day  Longitude
Latitude  \
0             100 -0.243530  2020-06-14 00:00:58   1        NaN
NaN
1             100 -0.215942  2020-06-14 00:01:58   1        NaN
NaN
2             100 -0.090149  2020-06-14 00:02:58   1        NaN
NaN
3             100 -0.089233  2020-06-14 00:03:58   1        NaN
NaN
4             100 -0.145142  2020-06-14 00:04:58   1        NaN
NaN
...           ...       ...                  ...  ..        ...
...
1529664        99 -1.079102  2020-06-20 23:55:40   1     104.16
```

特征提取

每7天提取一次数据，前一个数据和当前数据之差、均值、最大值、最小值、峰值、偏度等。

```
res = []
if flag == 'train':
    it = 130
else:
    it = 1
for i in tqdm(range(1, it+1)):
    day = gap*i
    tmp = df[(df['Day'] < day) & (df['Day'] >= (day-7))]

    tmp1 = df1[(df1['Day'] < day) & (df1['Day'] >= (day - 7))]
    train = pd.DataFrame()
    train['area'] = loc_df['area'].unique()
            dic = df_grp['average'].agg(opt).to_dict()
            col_name = 'sound_' + opt
            train[col_name] = train['area'].map(dic).values
        train['sound_max_min'] = train['sound_max'] - train['sound_min']

        for j in [1]:
            for opt in ['mean', 'max', 'min']:
                dic = df_grp[f'sound_diff_{j}'].agg(opt).to_dict()
                col_name = f'sound_diff_{j}_' + opt
                train[col_name] = train['area'].map(dic).values
        train['sound_diff_1_max_min'] = train['sound_diff_1_max'] - train['sound_d
        tmp = tmp[tmp['Day'] == day-1]
        df_grp = tmp.groupby('area')
        for opt in ['mean', 'max', 'min']:
            dic = df_grp['average'].agg(opt).to_dict()
            col_name = 'sound_' + opt + '_day'
            train[col_name] = train['area'].map(dic).values
        train['sound_max_min'] = train['sound_max_day'] - train['sound_min_day']

        for j in [1]:
            for opt in ['mean', 'max', 'min']:
                dic = df_grp[f'sound_diff_{j}'].agg(opt).to_dict()
                col_name = f'sound_diff_{j}_' + opt + '_day'
                train[col_name] = train['area'].map(dic).values
        train['sound_diff_1_max_min_day'] = train['sound_diff_1_max_day'] - train[
        df_grp = tmp1.groupby('area')
        for opt in ['mean',  'max', 'min']:
            dic = df_grp['average'].agg(opt).to_dict()
            col_name = 'mag_' + opt
            train[col_name] = train['area'].map(dic).values
        train['mag_max_min'] = train['mag_max'] - train['mag_min']
        for j in [1]:
            for opt in ['mean', 'max', 'min']:
                dic = df_grp[f'mag_diff_{j}'].agg(opt).to_dict()
                col_name = f'mag_diff_{j}_' + opt
```

一种基于特征排序的在线分配算法

特征过滤（correlation analysis）

averagesound_day_max_mean averagesound_day_min_mean averagesound_day_mean_max averagesound_day_mean_min sound_diff_1_day_max_mean sound_diff_1_day_min_mean sound_diff_1_day_mean_max sound_diff_1_day_mean_min averagemag_day_max_mean averagemag_day_min_mean averagemag_day_mean_max averagemag_day_mean_min mag_diff_1_day_max_mean mag_diff_1_day_min_mean mag_diff_1_day_mean_max mag_diff_1_day_mean_min sound_mean sound_max sound_min sound_max_min sound_diff_1_mean sound_diff_1_max sound_diff_1_min sound_diff_1_max_min sound_mean_day sound_max_day sound_min_day sound_diff_1_mean_day sound_diff_1_max_day sound_diff_1_min_day sound_diff_1_max_min_day mag_mean mag_max mag_min mag_max_min mag_diff_1_mean mag_diff_1_max mag_diff_1_min mag_diff_1_max_min mag_mean_day mag_max_day mag_min_day mag_diff_1_mean_day mag_diff_1_max_day mag_diff_1_min_day mag_diff_1_max_min_day

46个特征

'sound_mean_day',
'mag_diff_1_max_min_day', 'sound_min_day', 'mag_diff_1_day_max_mean',
'sound_diff_1_max_min', 'sound_diff_1_max_day', 'sound_min',
'mag_diff_1_max_day', 'sound_diff_1_min',

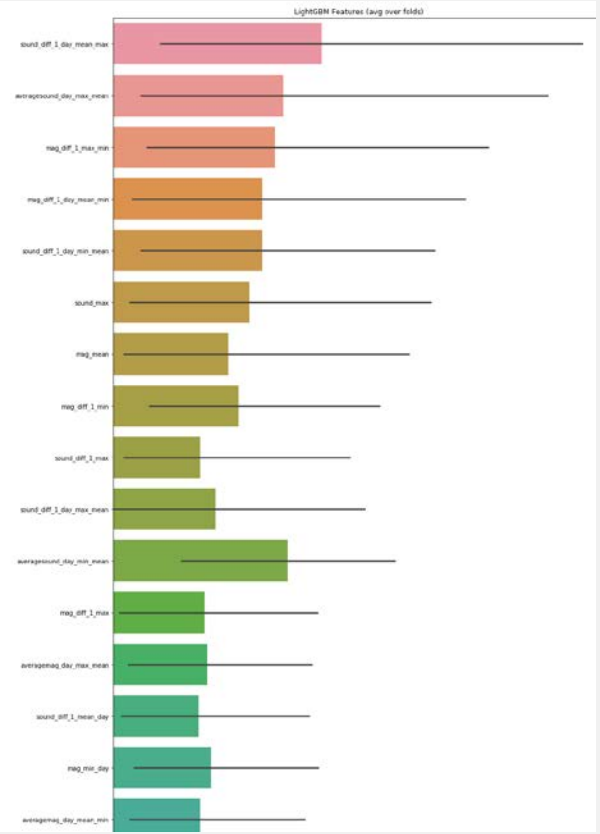
皮尔逊相关系数过滤掉10个特征

'averagesound_day_max_mean', 'averagesound_day_min_mean', 'averagesound_day_mean_max',
'averagesound_day_mean_min', 'sound_diff_1_day_max_mean', 'sound_diff_1_day_min_mean',
'sound_diff_1_day_mean_max', 'sound_diff_1_day_mean_min', 'averagemag_day_max_mean',
'averagemag_day_min_mean', 'averagemag_day_mean_max', 'averagemag_day_mean_min',
'mag_diff_1_day_mean_max', 'mag_diff_1_day_mean_min', 'sound_max', 'sound_max_min',
'sound_diff_1_mean', 'sound_diff_1_max', 'sound_max_day', 'sound_diff_1_mean_day',
'sound_diff_1_min_day', 'mag_mean', 'mag_min', 'mag_max_min', 'mag_diff_1_mean', 'mag_diff_1_max',
'mag_diff_1_min', 'mag_diff_1_max_min', 'mag_mean_day', 'mag_min_day', 'mag_diff_1_mean_day',
'mag_diff_1_min_day'

保留36个特征

特征过滤（LightGBM）

过滤掉权重小的4个特征，
最终保留32个特征



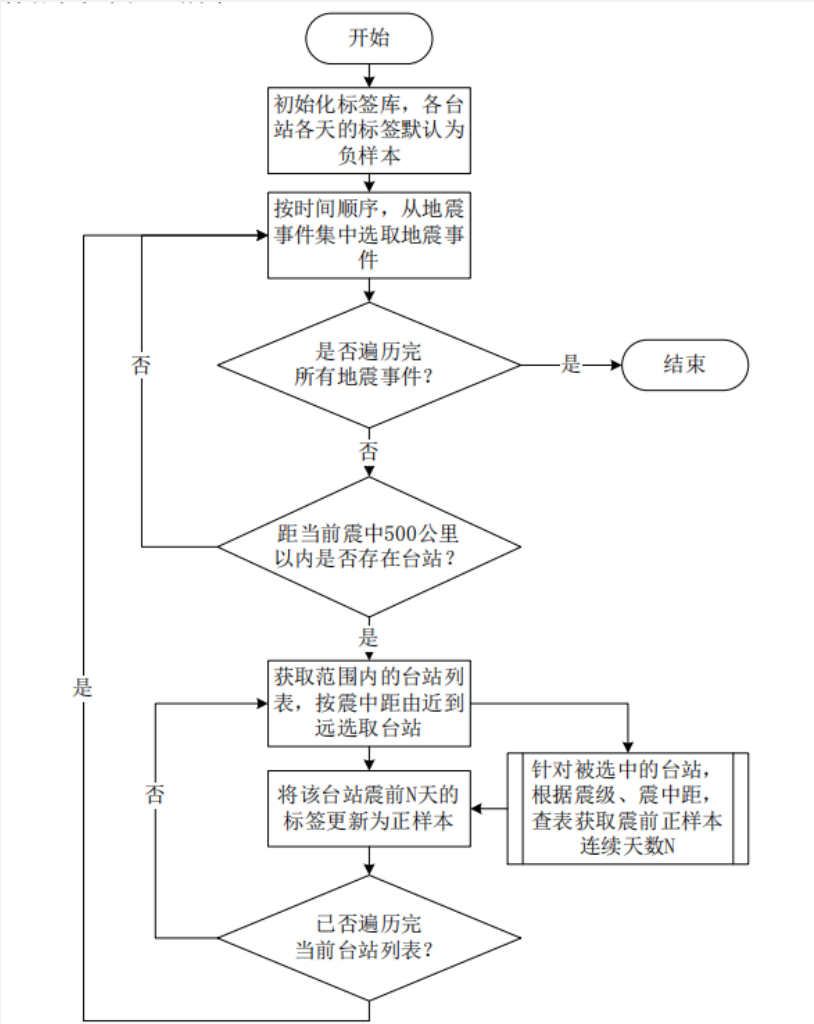
第三章

Part 03

样本集构建

样本集构建

台站震中距(KM)	弱震（0-3）	有感地震（3-4.5）	中强震（4.5-6）	强震（6以上）
<100	1天	3天	7天	15天
100-300	-	1天	3天	7天
300-500	-	-	1天	3天

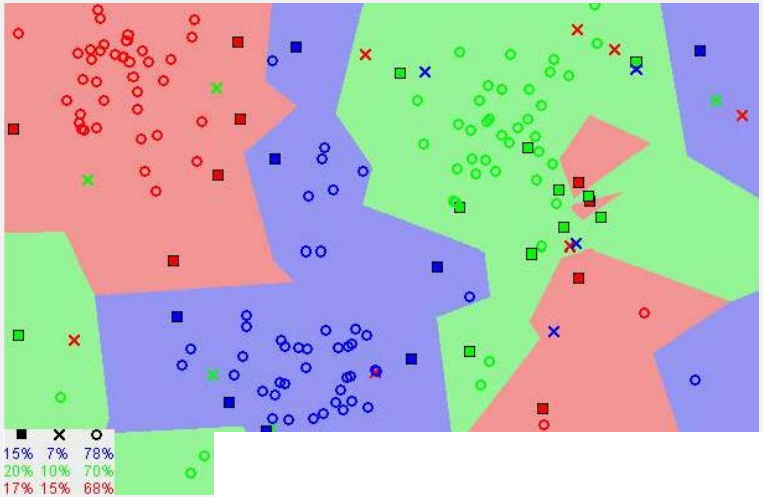


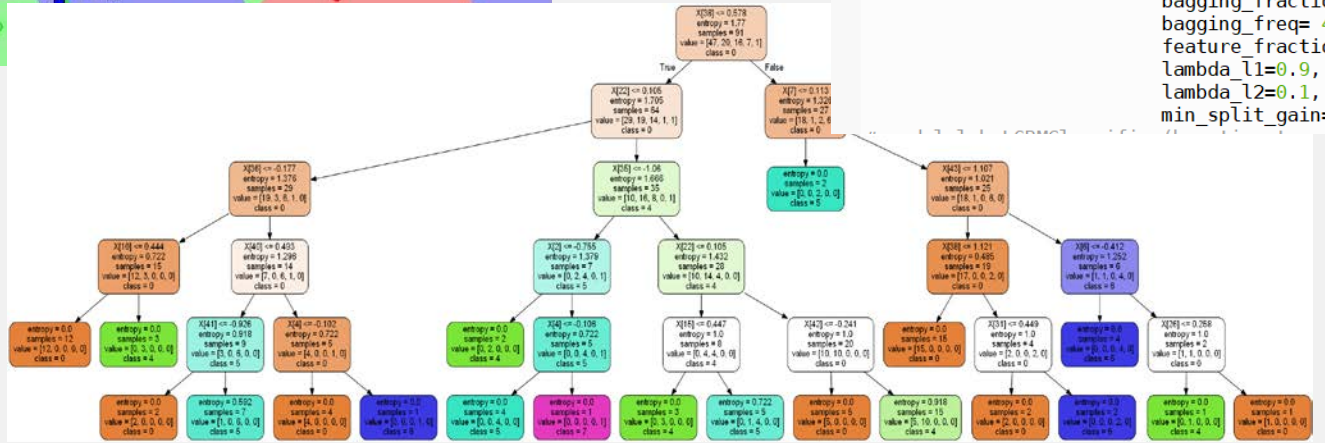
第四章

Part 04

预测模型

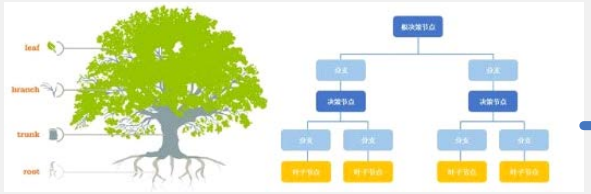
knn算法+LightGBM+决策树





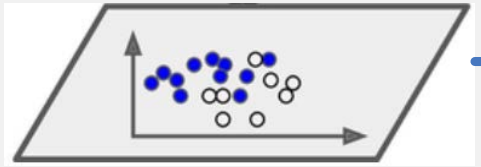
```
F=open(r'D:\di\A榜训练数据和测试数据\train_fe.pkl','rb')
train=pickle.load(F)
a = train[train['label'] != 0]
b = train[train['label'] == 0]
b = b.sample(frac=0.2, random_state=1)
train = a.append(b)
features = [x for x in train.columns if x not in ['label','weig
d = ['sound_mean_day', 'mag_diff_1_max_min_day',  'sound_min_da
for i in d:
    features.remove(i)
X=train[features].iloc[:,1:35]
y=train['label']
T=open(r'D:\di\final6\test_fe.pkl','rb')
test=pickle.load(T)
test1=test[features].iloc[:,1:]
model=lgb.LGBMClassifier(boosting_type='gbdt',
                         objective='binary',
                         learning_rate=0.01,#学习率
                         n_estimators=1000,#
                         max_depth=7,
                         num_leaves=95,
                         max_bin=255,
                         min_data_in_leaf=41,
                         bagging_fraction=1.0,
                         bagging_freq= 45,
                         feature_fraction= 0.6,
                         lambda_l1=0.9,
                         lambda_l2=0.1,
                         min_split_gain=0.5)
```

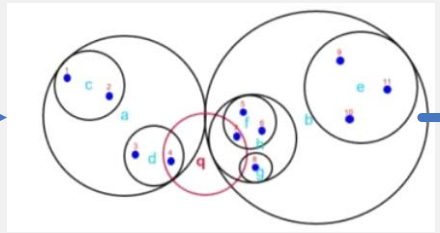
集成学习：权重自适应



弱学习器1

权重自适应

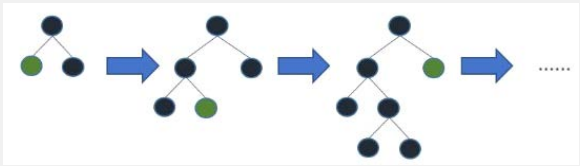




弱学习器2

Result

权重自适应



弱学习器3

感谢聆听