使用说明书

使用IDE PyCharm打开main.py，安装好所需的库模块（numpy，pandas，matplotlib，scikit-learn）之后直接运行即可。

也可在已经安装好相应的库模块的python环境中输入以下命令执行代码。

```
python main.py
```

设计说明书

首先，在拿到数据之后直接打开来观察数据，可以发现数据有许多缺失值，但通过 pandas 模块的统计方法发现只有其中三列是有较多缺失值的，并且相对于数据总量来说并没有达到不可用的地步。

```
(3276, 10)
ph                491
Hardness            0
Solids              0
Chloramines         0
Sulfate           781
Conductivity        0
Organic_carbon      0
Trihalomethanes   162
Turbidity           0
Potability          0
dtype: int64
```

因此，使用这一列的众数的平均值来填充缺失值。同时由于数据保留的小数位较多，因此计算众数的时候只取每个数据的整数部分。

```
    #  对于数值，均使用所在列的四舍五入后的值的众数进行
    df['ph'] = df['ph'].fillna(df['ph'].round().mode().mean())
    df['Sulfate'] = df['Sulfate'].fillna(df['Sulfate'].round().mode().mean())
    df['Trihalomethanes'] = df['Trihalomethanes'].\
        fillna(df['Trihalomethanes'].round().mode().mean())
```

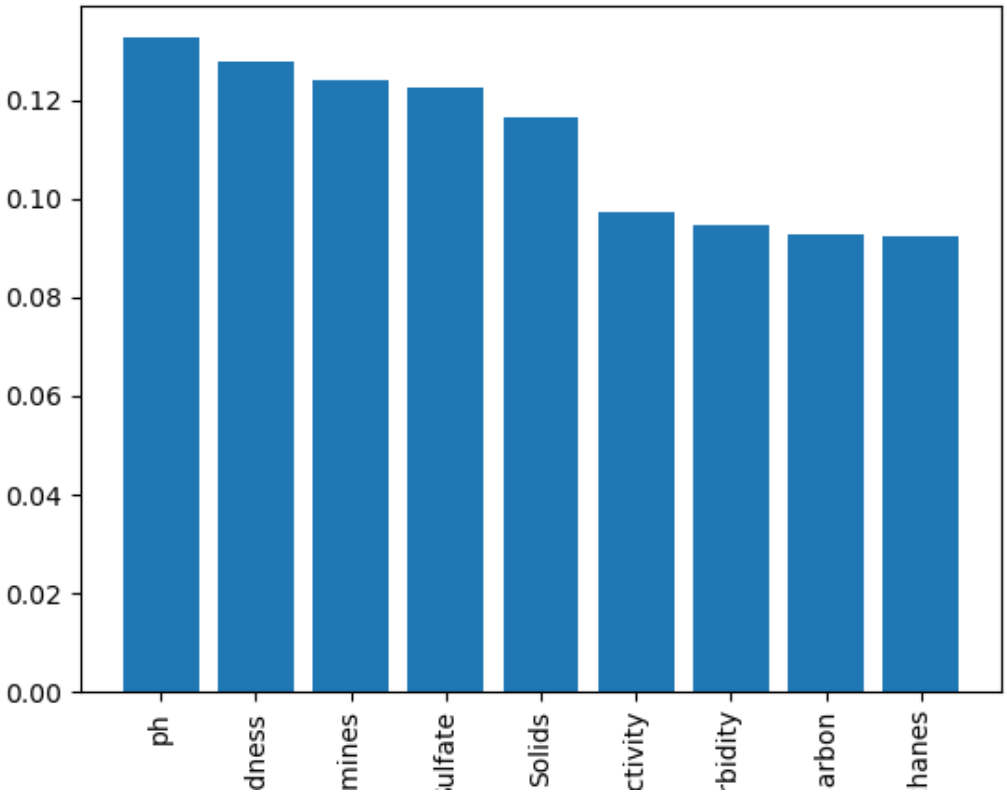
然后将预处理好的数据保存到本地。

经过打印观察，所有的特征都是数字，因此无需再进行额外的预处理。

然后开始使用随机森林拟合数据并探究每个特征的重要程度。

```
    #  由于特征值都为数字，因此无需再进行额外的处理，开始使用随机森林拟合并探究特征的重要性
    y = df.get("Potability")
    X = df.drop("Potability", axis=1)

    from sklearn.ensemble import RandomForestRegressor

    rf = RandomForestRegressor()
    rf.fit(X, y)

    feature_importances = rf.feature_importances_
    feature_importance_pairs = [(feature_name, feature_importance)
                                for feature_name, feature_importance in
                                zip(X.columns, feature_importances)]

    feature_importance_pairs = sorted(feature_importance_pairs, key=lambda x:
x[1], reverse=True)
    feature_importance_names = [name[0] for name in feature_importance_pairs]
    feature_importance_vals = [name[1] for name in feature_importance_pairs]

    import matplotlib.pyplot as plt
    figure = plt.figure()
    plt.bar(range(len(feature_importance_names)), feature_importance_vals,
orientation='vertical')
    plt.xticks(range(len(feature_importance_names)), feature_importance_names,
rotation='vertical')
    plt.show()
```

结果展示如下。经过上面的作图，我们可以发现，对于结果来说，九个特征均有一定的重要性，因此我们使用所有九个特征来进行训练拟合和预测，并在拟合训练完成后使用 joblib 模块的持久化方法将模型保存到本地。




```
    #  经过上述作图，我们可以发现对于结果来说，九个特征均有一定的重要性，因此我们使用所有九个特征进行预测
    from sklearn.model_selection import train_test_split

    train_X, test_X, train_y, test_y = train_test_split(X, y, test_size=0.25, random_state=114514)

    rf = RandomForestRegressor()
    model = rf.fit(train_X, train_y)

    import joblib
    joblib.dump(model, "./rf.pkl", compress=3)
```

然后对模型进行预测和评估。

```
    pred_y = model.predict(test_X)

    #  测试发现选取阈值为0.65时指标表现较好
    threshold = 0.65
    for i in range(len(pred_y)):
        if pred_y[i] >= threshold:
            pred_y[i] = 1
        else:
            pred_y[i] = 0

    #  模型评估
    from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
    accuracy_score_value = accuracy_score(test_y, pred_y)
    print(f"准确率:{accuracy_score_value}")
```

```
precision_score_value = precision_score(test_y, pred_y)
print(f"精确率:{precision_score_value}")

recall_score_value = recall_score(test_y, pred_y)
print(f"召回率:{recall_score_value}")

f1_score_value = f1_score(test_y, pred_y)
print(f"f1值:{f1_score_value}")
```

最后对所有的数据进行预测，并使用官方提供的 boto3 模块将预测结果写入S3存储。

```
#  将预测结果写入S3存储
pred_y_all = model.predict(X)
for i in range(len(pred_y_all)):
    if pred_y_all[i] >= threshold:
        pred_y_all[i] = 1
    else:
        pred_y_all[i] = 0

X['Potability'] = y
X['predict'] = pred_y_all
X.to_csv("./predict.csv", index=False)

import boto3
ACCESS_KEY = "B67DDB9A1DCDE1F2O8B4"
SECRET_KEY = "WzZGQ0MwOUEwNTlFNjI2RjgwMTkzQUZERkIwRDgy"
serviceEndpoint = "http://scut.depts.bingosoft.net:29997"
bucketName = 'luna'
fileName = "predict.csv"

print("Uploading result of predicting...")
s3 = boto3.client('s3',
                  aws_access_key_id=ACCESS_KEY,
                  aws_secret_access_key=SECRET_KEY,
                  endpoint_url=serviceEndpoint)
s3.upload_file(fileName, bucketName, fileName)
print("Upload success!")
```