

环境准备与基础知识

- Jupyter notebook

Jupyter Notebook是一块所见即所得的画布，通过在浏览器上编辑代码，让开发人员实现展示与快速迭代的利器。

更多内容参考：<https://www.jianshu.com/p/91365f343585>

- 决策树相关算法介绍

- <https://blog.csdn.net/qingqing7/article/details/78416708>

- 完成任务回顾

- 将任务分3块

1. 需求，目标
2. 确定方法，明确步骤，分步骤执行
3. 输出结果与调优

- 善用工具

- 如Bob（翻译插件）、Dash（API文档）

- 勤于总结

- pd、np常用操作等

- 积极沟通

第一次机器学习内容

Lab02 - Decision Tree - I

1、案例：泰坦尼克号幸存者预测

案例结构：

1. 引入依赖

```
# 数据分析和准备
import pandas as pd
import numpy as np
```

```
import random as rnd

# 可视化
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

# 机器学习
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
```

2. 读取数据（训练数据和测试数据）

```
# 读取数据文件到dataframe
train_df = pd.read_csv('train.csv')

# 查看前5行
train_df.head()

# 查看数据类型和缺失值
train_df.info()
```

3. 人为分析数据（数据概况、缺失值等）

```
# 输出表头信息
print(train_df.columns.values.tolist())

# 根据百分比分布查看数据
train_df.describe(percentiles=[.1, .2, .3, .4, .6, .7, .8, .9, .95,
                                .98, .99])

# 查看连续特征的数据列分析
train_df.describe()

# 查看分类特征的数据列分析
train_df.describe(include=[ 'O' ])
```

4. 数据分析，清洗数据（相关性分析，缺失值填充等）

```
# 众数填充空值
```

```
df['Embarked'].fillna(df['Embarked'].mode(dropna=True)[0],  
inplace=True)
```

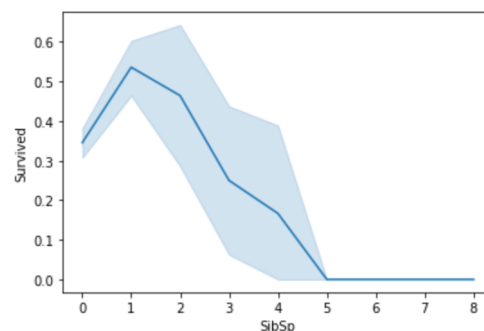
```
# 数据表格 数据替换 (数字编码分类特征)
```

```
num_encode = {  
    'Sex': {'male': 0, "female": 1},  
    'Embarked': {'S': 0, 'Q': 1, 'C': 2 }  
}  
df.replace(num_encode, inplace=True)
```

5. 可视化输出, 方便分析

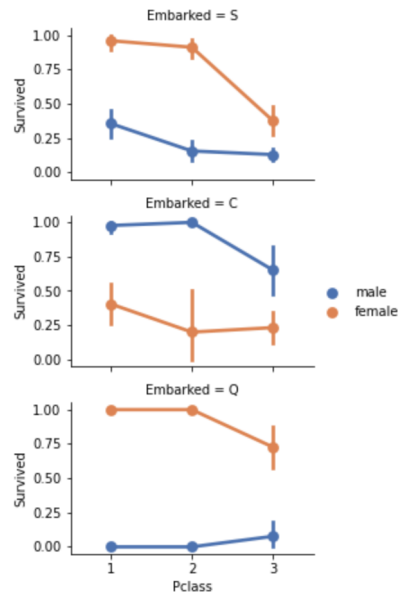
```
# 折线图
```

```
sns.lineplot(x='SibSp', y='Survived', data=train_df)
```



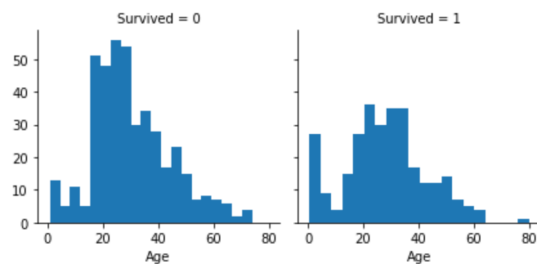
```
# 格点图
```

```
grid = sns.FacetGrid(train_df, row='Embarked', height=2.2,  
aspect=1.6)  
grid.map(sns.pointplot, 'Pclass', 'Survived', 'Sex',  
palette='deep')  
grid.add_legend()
```



柱状图

```
g = sns.FacetGrid(train_df, col='Survived')
g.map(plt.hist, 'Age', bins=20)
```



6. 建模、预测和评价

切分数据集，用于训练和验证

```
x_train, x_test, y_train, y_test = train_test_split(X, y,
test_size=0.3)
```

决策树建模和训练

```
tree = DecisionTreeClassifier(criterion='entropy')
tree.fit(x_train, y_train)
```

决策树预测

```
y_pred = tree.predict(x_test)
```

评价，输出准确度

```
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
```

2、任务：气缸带数据集band type预测

任务结构:

1. 引入依赖包

依然使用决策树算法, 引入包同上案例

2. 导入数据

```
# 指定列名和缺失值导入数据到dataframe
names = ['c1', 'c2', 'c3']
missing_values = ["?", "nan"]
train_df =
pd.read_csv('bands_init.data', names=names, na_values=missing_values)
```

3. 清洗数据

```
# 遍历替换
# 把df中的 ? 替换成 'nan'
# 把df中 str 类型的值全部转换成小写
temp = 'aa22'
feature_list = train_df.columns.tolist()
for index in train_df.index:
    for feature in feature_list:
        if type(train_df.loc[index, feature]) == type(temp):
            train_df.loc[index, feature] = train_df.loc[index,
feature].lower()
        if train_df.loc[index, feature] == '?':
            # print(train_df.loc[index, feature])
            train_df.loc[index, feature] = np.nan
```

```
# 列类型转换
# 从20列到39列 将列属性转换成float类型
i=20
while(i<39):
    count = 0
    for row in train_df[feature_list[i]]:
        try:
            float(row)
        except ValueError:
            train_df.loc[count, feature_list[i]] = np.nan
            pass
        count=count+1
```

```

train_df[feature_list[i]]=train_df[feature_list[i]].astype(np.float64)
# 输出转换过的列名
print(feature_list[i])
i=i+1

```

4. 数据分析

同上案例

5. 数据可视化

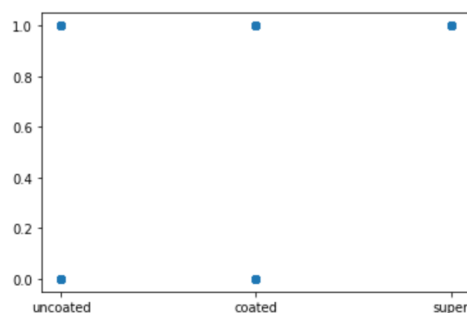
画图时，当数据有空值会导致画图失败。

解决办法：先填充空值再画图分析。

```

# 散点图
x = train_df['paper type']
y = train_df['band type']
color = ['r', 'y', 'k', 'g', 'm']
plt.scatter(x, y)
plt.show()

```



6. 再次清洗数据

```

# 删除目标值为空的数据（错误数据（如出现错行现象））
for index in range(train_df.shape[0]):
    try:
        if np.isnan(train_df['band type'][index]):
            train_df=train_df.drop([index])
    except:
        pass

```

7. 准备数据

可以把encode的map定义大些，X可以从中选取几个训练，方便调整训练模型。

用大map数据替换

```
encode = {
    'grain screened':{'yes':0,'no':1},
    'proof on ctd ink':{'yes':0,'no':1},
    'blade mfg':{'uddeholm':0,'benton':1},
    'paper type':{'coated':0,'uncoated':1,'super':2},
    'ink type':{'coated':0,'uncoated':1,'cover':2},
    'direct steam':{'yes':0,'no':1},
    'type on cylinder':{'yes':0,'no':1},
    'plating tank':{'1911':0,'1910':1},
    'solvent type':{'naptha':0,'line':1,'xylol':2},
    'press type':
    {'mott70':0,'mott94':1,'albert70':2,'woodhoe70':3},
    'paper mill location':
    {'scandinavian':0,'canadian':1,'northus':2,'mideuropean':3,'southus':4}
}
train_df.replace(encode, inplace=True)

# 选取不同列作为训练数据集，之后进行训练
X = train_df[['grain screened','blade mfg','direct steam','paper type', 'ink type','press type','solvent type']]
# X = train_df[['grain screened','blade mfg','direct steam','type on cylinder','plating tank']]
y = train_df['band type']
```

8. 建模、预测和评价

同上