Python与大数据分析

--数据分析

知识点概括

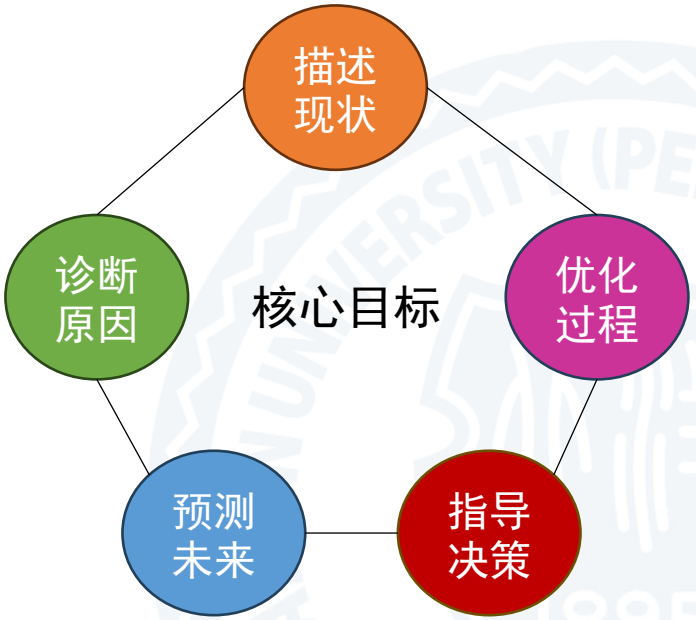
- 数据分析概述
- 泰坦尼克号项目全流程

数据分析

数据分析（Data Analysis） 是通过系统性方法对数据进行整理、转换、建模和解释，以提取有用信息、发现潜在规律，并支持决策的过程。

核心目标：

1. 描述现状：通过对现有数据的分析，总结出当前的状况或模式。
2. 诊断原因：探究问题或现象背后的原因。
3. 预测未来：利用数据模型对未来的发展趋势进行预测。
4. 指导决策：根据数据分析结果，为决策提供依据。
5. 优化过程：挖掘潜在的改进空间，通过数据驱动优化现有流程或系统。



数据分析

以下为典型机器学习数据分析项目的通用流程：

1. 问题定义

- 明确目标（如分类、回归、聚类）。
- 确定评估指标（如准确率、F1分数、RMSE）

2. 数据收集与理解

- 数据来源：数据库、API、日志文件、公开数据集等。
- 数据探索（EDA）：
- 统计分布分析（均值、方差、分位数）。
- 可视化分析（直方图、散点图、箱线图）。
- 缺失值、异常值检测。

3. 数据预处理

- **清洗：**
  - 处理缺失值：删除、插值、模型预测。
  - 处理异常值：截断、分箱、统计检验。
- **转换：**
  - 标准化、归一化。
  - 类别编码：独热编码、标签编码。
- **特征工程：**
  - 特征构造（如从日期提取星期、月份）。
  - 特征选择：过滤法（卡方检验）、包裹法（递归特征消除）、嵌入法（L1正则化）。
  - 降维：PCA、LDA（线性判别分析）。

数据分析

4. 模型选择与训练

- **划分数据集**：训练集、验证集、测试集（比例通常为60-20-20或70-30）。
- **选择模型**：
  - 简单问题：线性模型、决策树。
  - 复杂问题：集成模型、深度学习。
- **训练与验证**：
  - 交叉验证（Cross-Validation）：K折交叉验证。
  - 超参数调优：网格搜索（Grid Search）、随机搜索（Random Search）、贝叶斯优化。

5. 模型评估

- **分类任务**：
  - 准确率、精确率、召回率、F1分数、ROC-AUC。
- **回归任务**：
  - 均方误差（MSE）、平均绝对误差（MAE）、R^2分数。
- **聚类任务**：
  - 轮廓系数（Silhouette Score）、Calinski-Harabasz指数。

后续以kaggle上面一个经典的比赛 ”Titanic - Machine Learning from Disaster”为例介绍一下数据分析全流程

泰坦尼克号数据分析流程

比赛概述

泰坦尼克号的沉没是历史上最臭名昭著的沉船事件之一。1912 年 4 月 15 日，在她的首次航行中，被广泛认为“永不沉没”的泰坦尼克号在与冰山相撞后沉没。不幸的是，没有足够的救生艇供船上每个人使用，导致 2224 名乘客和船员中有 1502 人死亡。

虽然幸存中涉及到一些运气成分，但似乎有些人比其他人更有可能幸存。

在这个比赛中，要求使用乘客数据（即姓名、年龄、性别、社会经济阶层等），建立一个预测模型来回答以下问题：“什么样的人更有可能幸存？”

Titanic - Machine Learning from Disaster 比赛地址： https://www.kaggle.com/competitions/titanic

泰坦尼克号数据分析流程

数据介绍

有两个相似的数据集，一个是训练集**train.csv**，另一个是测试集**test.csv**：

- **train.csv**包含乘客子集的详细信息（准确地说是 891 人），揭示了他们是否幸存，也称为“基本事实”。
- **test.csv** 数据集包含类似的信息，但没有透露每位乘客是否幸存，预测这些结果是你的工作，即：使用你在 train.csv 数据中找到的模式，预测船上的其他 418 名乘客（在 test.csv 中找到）是否幸存

泰坦尼克号数据分析流程

下面了解一下train.csv数据集中的变量：

变量	定义	取值
PassengerId	乘客编号	1，2，…，891
Survived	幸存与否	0=死亡，1=幸存
Pclass	票价等级	1=一等票，2=二等票，3=三等票
Name	乘客姓名	字符型数据，取值均不同
Sex	乘客性别	male=男性，female=女性
Age	乘客年龄	0-80岁，有缺失值
SibSp	在船兄弟姐妹或配偶数量	0-8个，无缺失值
Parch	在船父母或孩子数量	0-6个，无缺失值
Ticket	票号	字符数值型混合数据，有重复值
Fare	票价	0-512美元
Cabin	客舱号	混合数据，有重复值，有缺失值
Embarked	登船港口	C=瑟堡，Q=皇后镇，S=南安普顿

泰坦尼克号数据分析流程

（1）导入包

```
import warnings
warnings.filterwarnings("ignore") #忽略警告信息

# 数据处理清洗包
import pandas as pd
import numpy as np
import random as rnd

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

# 机器学习算法相关包
from sklearn.linear_model import LogisticRegression, Perceptron, SGDClassifier
from sklearn.svm import SVC, LinearSVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
```

泰坦尼克号数据分析流程

（2）加载数据集

```
train_df = pd.read_csv('titanic/train.csv')
test_df = pd.read_csv('titanic/test.csv')
combine = [train_df, test_df] # 合并数据
train_df # 查看数据
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows × 12 columns

泰坦尼克号数据分析流程

（3）描述性统计分析

```
# 获取所有特征名
print(train_df.columns.values)

# 查看缺失值（空值，null,nan)
print(train_df.isnull().sum())
print(test_df.isnull().sum())
```

```
['PassengerId' 'Survived' 'Pclass' 'Name' 'Sex' 'Age' 'SibSp' 'Parch'
 'Ticket' 'Fare' 'Cabin' 'Embarked']
```

训练集

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

测试集

```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

哪些特征包含缺失值（空值，null，nan)？

- 在训练集中，缺失值数目 Cabin > Age > Embarked；
- 在测试集中，缺失值数目 Cabin > Age > Fare。

泰坦尼克号数据分析流程

（3）描述性统计分析

```
# 查看各个特征的数据类型
train_df.info()
test_df.info()
```

各种特征的数据类型是什么？

- 在训练集中，7个特征是整数型或浮点型，5个特征是字符串型；
- 在测试集中，6个特征是整数型或浮点型，5个特征是字符串型。

训练集

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

测试集

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Pclass       418 non-null    int64
 2   Name         418 non-null    object
 3   Sex          418 non-null    object
 4   Age          332 non-null    float64
 5   SibSp        418 non-null    int64
 6   Parch        418 non-null    int64
 7   Ticket       418 non-null    object
 8   Fare         417 non-null    float64
 9   Cabin        91 non-null     object
 10  Embarked     418 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

泰坦尼克号数据分析流程

（3）描述性统计分析

```
# 查看数值特征的分布
round(train_df.describe(percentiles=[.5, .6, .7, .75, .8, .9, .99]),2)
```

样本中数值特征的分布是什么？

- 样本总数为891 人，约占泰坦尼克号上实际乘客人数（2,224 人）的40%；
- Survived 是具有 0 或 1 值的二分类变量，并且大约 38% 的样本存活，代表实际存活率32%；
- 大多数乘客（>50%)的票价等级是三等票；
- 年龄在 65-80 岁之间的老年乘客很少 (<1%)；
- 近 30% 的乘客有兄弟姐妹或配偶一同登机；
- 大多数乘客 (>75%) 没有与父母或孩子一起旅行；
- 票价差异很大，少数乘客 (<1%) 支付高达 512 美元。

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.00	891.00	891.00	714.00	891.00	891.00	891.00
mean	446.00	0.38	2.31	29.70	0.52	0.38	32.20
std	257.35	0.49	0.84	14.53	1.10	0.81	49.69
min	1.00	0.00	1.00	0.42	0.00	0.00	0.00
50%	446.00	0.00	3.00	28.00	0.00	0.00	14.45
60%	535.00	0.00	3.00	31.80	0.00	0.00	21.68
70%	624.00	1.00	3.00	36.00	1.00	0.00	27.00
75%	668.50	1.00	3.00	38.00	1.00	0.00	31.00
80%	713.00	1.00	3.00	41.00	1.00	1.00	39.69
90%	802.00	1.00	3.00	50.00	1.00	2.00	77.96
99%	882.10	1.00	3.00	65.87	5.00	4.00	249.01
max	891.00	1.00	3.00	80.00	8.00	6.00	512.33

泰坦尼克号数据分析流程

（4）基于数据分析的假设

我们想知道每个特征与幸存Survived的相关性如何，以便后期建模。事先可以假设：

- Age 年龄特征肯定与幸存相关;
- Embarked 登船港口可能与幸存或其他重要特征相关；
- Ticket 票号包含较高重复率(22%)，并且和幸存之间可能没有相关性，因此可能会从我们的分析中删除；
- Cabin 客舱号可能被丢弃，因为它在训练和测试集中缺失值过多（数据高度不完整）；
- PassengerId 乘客编号可能会从训练数据集中删除，因为它对幸存没有作用；
- Name 名字特征比较不规范，可能也对幸存没有直接贡献，因此可能会被丢弃。

泰坦尼克号数据分析流程

（4）基于数据分析的假设

```
# 针对Pclass和Survived进行分类汇总
train_df[['Pclass','Survived']].groupby(['Pclass'],as_index=False).mean().sort_values(by='Survived', ascending=False)
```

	Pclass	Survived
0	1	0.629630
1	2	0.472826
2	3	0.242363

Pclass 观察到 Pclass=1 和 Survived 之间有显著的相关性（>0.5），因此在模型中应包含此特征

泰坦尼克号数据分析流程

（4）基于数据分析的假设

```
# 针对Sex和Survived进行分类汇总
train_df[['Sex','Survived']].groupby(['Sex'],as_index=False).mean().sort_values(by='Survived',
ascending=False)
```

	Sex	Survived
0	female	0.742038
1	male	0.188908

Sex 女性的存活率非常高达 74%，因此在模型中应包含性别特征

泰坦尼克号数据分析流程

（4）基于数据分析的假设

```
# 针对SibSp和Survived进行分类汇总
train_df[['SibSp','Survived']].groupby(['SibSp'],as_index=False).mean().sort_values(by='Survived',ascending=False)
```

	SibSp	Survived
1	1	0.535885
2	2	0.464286
0	0	0.345395
3	3	0.250000
4	4	0.166667
5	5	0.000000
6	8	0.000000

SibSp 这个特征对于某些值与Survived具有零相关性，最好从这些单独的特征中派生一个特征或一组特征，使得与Survived有显著相关性

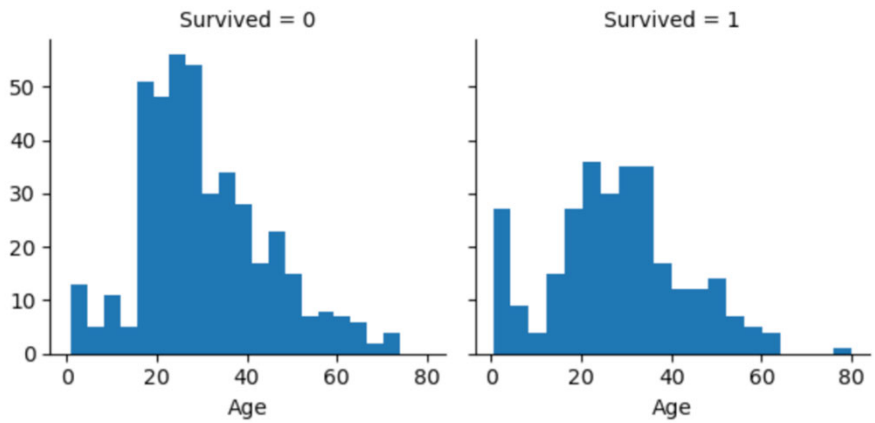
泰坦尼克号数据分析流程

（5）可视化数据分析

分析数值特征 Age 与 Survived 相关性

首先了解数值特征Age与我们的解决方案目标（Survived）之间的相关性。直方图对于分析连续数值变量很有用。

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



- 婴儿（年龄<=4）的存活率很高；
- 最年长的乘客（年龄 = 80）幸存下来；
- 大部分15-25 岁的人无法生存；
- 大多数乘客的年龄在 15-35 岁之间。

这个简单的分析证实了后续工作我们应该：

(1)在模型训练中考虑年龄特征 Age；　(2)完成年龄特征的缺失值处理；　(3)捆绑年龄组。

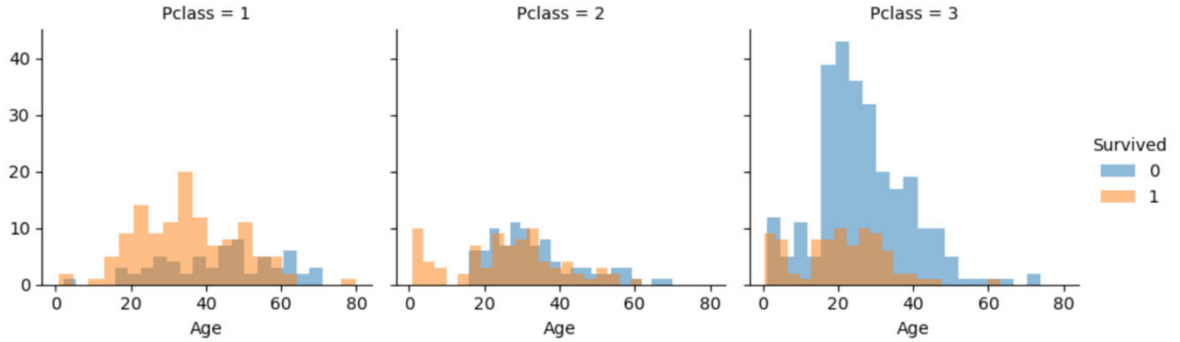
泰坦尼克号数据分析流程

（5）可视化数据分析

```
#分析有序分类特征 Pclass 与 Survived 相关性
grid = sns.FacetGrid(train_df, col='Pclass', hue='Survived') #颜色分组（hue="Survived"）在每个子图中，用不同颜色表示存活状态。
grid.map(plt.hist, 'Age', alpha=0.5, bins=20)
grid.add_legend();
```

- Pclass=3 有大多数乘客，但大多数没有幸存；
- Pclass=2 和 Pclass=3 的婴儿乘客大多幸存下来；
- Pclass=1 中的大多数乘客幸存下来；
- Pclass 因乘客年龄分布而异。

综上分析，应该将Pclass纳入模型训练之中。



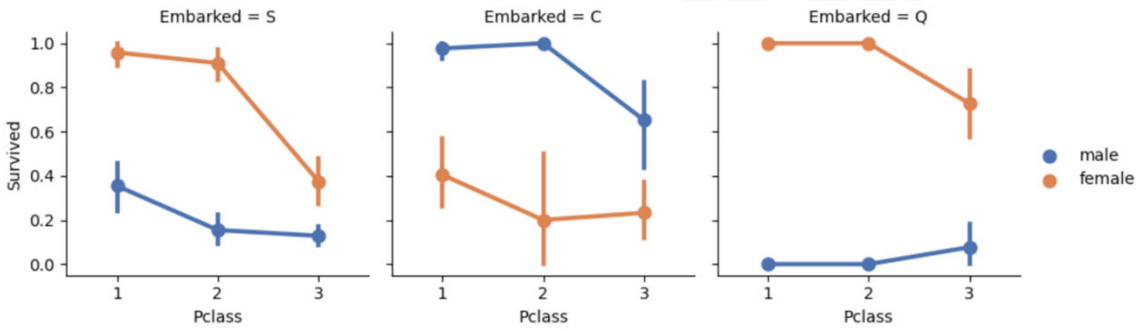
泰坦尼克号数据分析流程

（5）可视化数据分析

分析数值特征 Fare 与 Survived 相关性

```
grid = sns.FacetGrid(train_df, col='Pclass', hue='Survived')
grid.map(plt.hist, 'Age', alpha=0.5, bins=20)
grid.add_legend();
```

- Embarked=S 和 Q 中，女性乘客的存活率远高于男性；Embarked=C 中，男性的存活率较高。这可能是 Embarked 和 Sex 相关，而 Sex 和 Survived 相关，进而造成Embarked与Survived间接相关；
- 对于同一票价等级和同一性别，不同登船港口的存活率不同。



综上分析，在模型训练中应该：添加 Sex 和 Embarked 特征。

泰坦尼克号数据分析流程

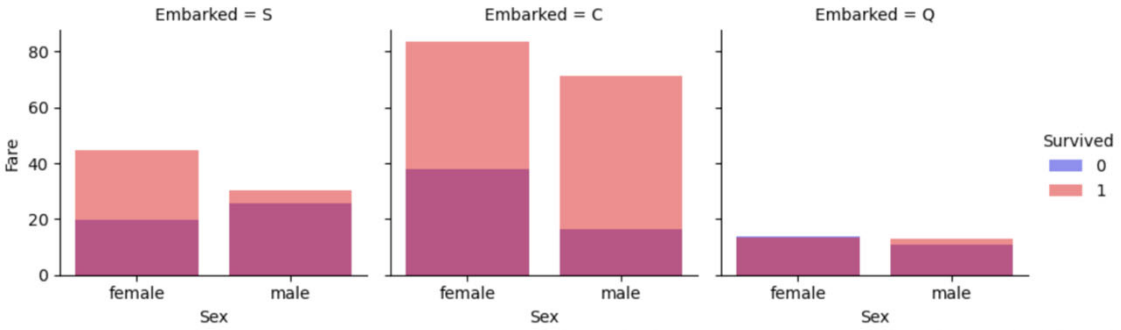
（5）可视化数据分析

分析数值特征 Age 与 Survived 相关性

将分类特征（具有非数值）和数字特征相关联，可以考虑将 Embarked（分类非数值）、Fare（连续数值）与 Survived（二分类数值）相关联。

```
grid = sns.FacetGrid(train_df, col='Embarked', hue='Survived', palette={0: 'b', 1: 'r'})
grid.map(sns.barplot, 'Sex', 'Fare', alpha=.5, ci=None)
grid.add_legend()
```

- 支付更高票价的乘客能够更好地幸存，并且幸存率具有较明显的票价区间性；
- 不同的登船港口有不同的存活率。



因此，应该考虑捆绑票价特征Fare，并纳入模型训练中。

泰坦尼克号数据分析流程

（6）整理、清洗数据

删除无用特征 Ticket 和 Cabin

注意一起删除训练集和测试集合的无用特征

```
print("Before", train_df.shape, test_df.shape, combine[0].shape, combine[1].shape)

train_df = train_df.drop(['Ticket', 'Cabin'], axis=1)
test_df = test_df.drop(['Ticket', 'Cabin'], axis=1)
combine = [train_df, test_df]

print("After", train_df.shape, test_df.shape, combine[0].shape, combine[1].shape)
```

```
Before (891, 12) (418, 11) (891, 12) (418, 11)
After (891, 10) (418, 9) (891, 10) (418, 9)
```

泰坦尼克号数据分析流程

（6）整理、清洗数据

从现有特征中提取新特征

在删除 Name 和 PassengerId 特征之前，分析是否可以设计 Name 特征来提取头衔 Title，并测试Title和Survived之间的相关性；

使用正则表达式提取Title特征。正则表达式(\w+\.)匹配 Name 特征中以点字符结尾的第一个单词，expand=False返回一个 DataFrame。

```
# 使用正则表达式提取Title特征
for dataset in combine:
        dataset['Title'] = dataset.Name.str.extract('([A-Za-z]+)\.', expand=False)

# pd.crosstab列联表
pd.crosstab(train_df['Title'], train_df['Sex']).sort_values(by='female', ascending=False)
```

Sex	female	male
Title		
Miss	182	0
Mrs	125	0
Mlle	2	0
Mme	1	0
Countess	1	0
Dr	1	6
Ms	1	0
Lady	1	0
Capt	0	1
Rev	0	6
Mr	0	517
Master	0	40
Col	0	2
Major	0	2
Jonkheer	0	1
Don	0	1
Sir	0	1

泰坦尼克号数据分析流程

（6）整理、清洗数据

从现有特征中提取新特征

```
# 可以用更常见的名称替换许多标题或将它们归类为稀有
for dataset in combine:
        dataset['Title'] = dataset['Title'].replace(['Lady', 'Countess','Capt', 'Col','Don', 'Dr', 'Major', 'Rev',
'Sir', 'Jonkheer', 'Dona'], 'Rare')
        dataset['Title'] = dataset['Title'].replace(['Mlle', 'Ms'], 'Miss')
        dataset['Title'] = dataset['Title'].replace('Mme', 'Mrs')

train_df[['Title', 'Survived']].groupby(['Title'], as_index=False).mean()
```

	Title	Survived
0	Master	0.575000
1	Miss	0.702703
2	Mr	0.156673
3	Mrs	0.793651
4	Rare	0.347826

泰坦尼克号数据分析流程

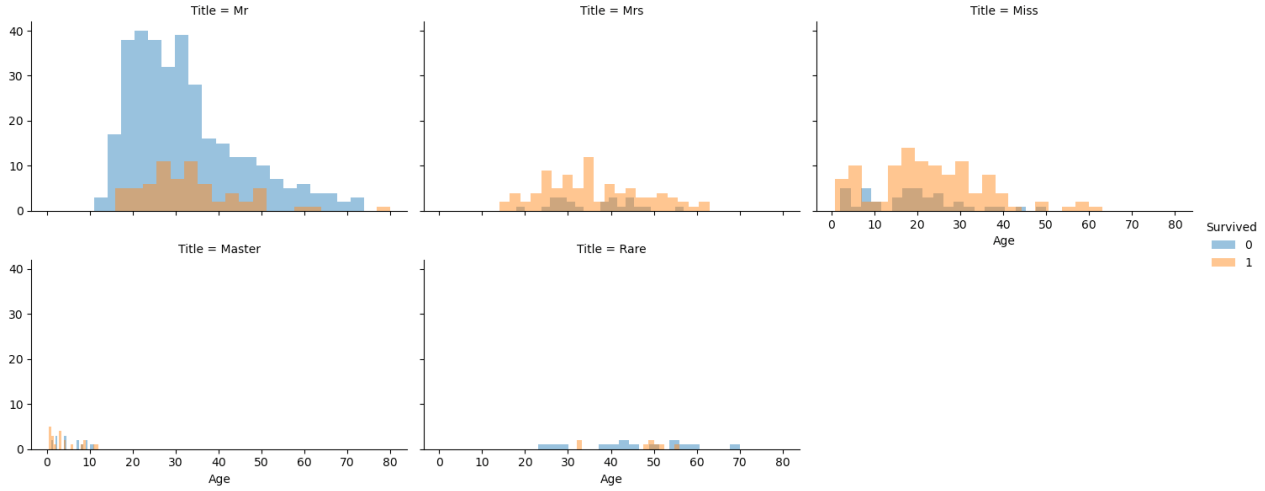
（6）整理、清洗数据

从现有特征中提取新特征

```
grid = sns.FacetGrid(train_df, col='Title', hue='Survived', col_wrap=3, aspect=1.6)
grid.map(plt.hist, 'Age', alpha=0.5, bins=20)
grid.add_legend()
```

- 不同的TiTle有不同的存活率。
- 基本在所有年龄段，TiTle为Mrs、Miss的存活率比TiTle为Mr的高，应该与Sex相关。

因此，应该考虑TiTle纳入模型训练中。



泰坦尼克号数据分析流程

（6）整理、清洗数据

从现有特征中提取新特征

```
# 将分类标题转换为序数
title_mapping = {"Mr": 1, "Miss": 2, "Mrs": 3, "Master": 4, "Rare": 5}
for dataset in combine:
        dataset['Title'] = dataset['Title'].map(title_mapping
        dataset['Title'] = dataset['Title'].fillna(0)

train_df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Fare	Embarked	Title
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	7.2500	S	1
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	71.2833	C	3
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	7.9250	S	2
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	53.1000	S	3
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	8.0500	S	1

泰坦尼克号数据分析流程

（6）整理、清洗数据

从现有特征中提取新特征

```
# 现在可以从训练和测试数据集中删除Name特征以及训练集中的
PassengerId 特征
train_df = train_df.drop(['Name', 'PassengerId'], axis=1)
test_df = test_df.drop(['Name'], axis=1)
combine = [train_df, test_df]
train_df.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked	Title
0	0	3	male	22.0	1	0	7.2500	S	1
1	1	1	female	38.0	1	0	71.2833	C	3
2	1	3	female	26.0	0	0	7.9250	S	2
3	1	1	female	35.0	1	0	53.1000	S	3
4	0	3	male	35.0	0	0	8.0500	S	1

泰坦尼克号数据分析流程

（6）整理、清洗数据

转换性别特征Sex

```
for dataset in combine:
#男性赋值为0，女性赋值为1，并转换为整型数据
dataset['Sex'] = dataset['Sex'].map( {'female': 1, 'male': 0} ).astype(int)

train_df.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked	Title
0	0	3	0	22.0	1	0	7.2500	S	1
1	1	1	1	38.0	1	0	71.2833	C	3
2	1	3	1	26.0	0	0	7.9250	S	2
3	1	1	1	35.0	1	0	53.1000	S	3
4	0	3	0	35.0	0	0	8.0500	S	1

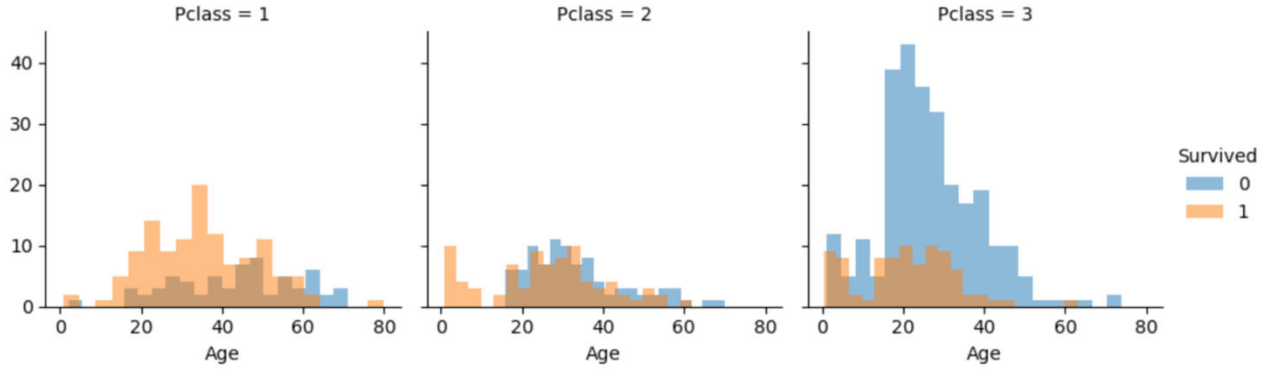
泰坦尼克号数据分析流程

（6）整理、清洗数据

填补年龄特征Age的缺失值

- 法一：使用其他相关特征，由前面的分析可知 Age、Sex 和 Pclass 之间的有相关性，故可使用 Pclass 和 Sex 特征组合集的 Age 中值预测 Age 值；
- 法二：基于法一，使用基于 Pclass 和Sex 组合集的均值和标准差之间的随机数来预测 Age 值；
- 法三：引入随机噪声。

我们用法一去做。



泰坦尼克号数据分析流程

（6）整理、清洗数据

```
# 遍历 Sex (0 或 1) 和 Pclass (1, 2, 3) 来计算六种组合的 Age 猜测值
for dataset in combine:
    dataset['Sex'] = dataset['Sex'].map({'male': 0, 'female': 1}).astype(int)
    dataset['Pclass'] = dataset['Pclass'].astype(int)
# 计算全局中位数备用
global_age_median = train_df['Age'].median()
for dataset in combine:
    guess_ages = np.zeros((2, 3))
    for i in range(0, 2):
        for j in range(0, 3):
            guess_df = dataset[(dataset['Sex'] == i) & (dataset['Pclass'] == j+1)]['Age'].dropna()
            age_guess = guess_df.median() if not guess_df.empty else global_age_median
            guess_ages[i,j] = int(age_guess / 0.5 + 0.5) * 0.5

    for i in range(0, 2):
        for j in range(0, 3):
            cond = (dataset['Age'].isnull()) & (dataset['Sex'] == i) & (dataset['Pclass'] == j+1)
            dataset.loc[cond, 'Age'] = guess_ages[i,j]

    dataset['Age'] = dataset['Age'].fillna(global_age_median).astype(int)
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Sex       891 non-null    int64
 3   Age       891 non-null    int64
 4   SibSp     891 non-null    int64
 5   Parch     891 non-null    int64
 6   Fare      891 non-null    float64
 7   Embarked  889 non-null    object
 8   Title     891 non-null    int64
dtypes: float64(1), int64(7), object(1)
memory usage: 62.8+ KB
```

无缺失值

泰坦尼克号数据分析流程

（6）整理、清洗数据

创建年龄段,并确定其与Survived的相关性。一般在建立分类模型时，需要对连续变量离散化，特征离散化后，模型会更稳定，降低了模型过拟合的风险。

```
# 将年龄分割为5段,等距分箱
train_df['AgeBand'] = pd.cut(train_df['Age'], 5)
train_df[['AgeBand', 'Survived']].groupby(['AgeBand'],
as_index=False).mean().sort_values(by='AgeBand', ascending=True)
```

	AgeBand	Survived
0	(-0.08, 16.0]	0.550000
1	(16.0, 32.0]	0.337374
2	(32.0, 48.0]	0.412037
3	(48.0, 64.0]	0.434783
4	(64.0, 80.0]	0.090909

泰坦尼克号数据分析流程

（6）整理、清洗数据

```
# 将这些年龄区间替换为序数
for dataset in combine:
    dataset.loc[ dataset['Age'] <= 16, 'Age'] = 0
    dataset.loc[(dataset['Age'] > 16) & (dataset['Age'] <= 32), 'Age'] = 1
    dataset.loc[(dataset['Age'] > 32) & (dataset['Age'] <= 48), 'Age'] = 2
    dataset.loc[(dataset['Age'] > 48) & (dataset['Age'] <= 64), 'Age'] = 3
    dataset.loc[ dataset['Age'] > 64, 'Age'] = 4

# 删除训练集中的AgeBand特征
train_df = train_df.drop(['AgeBand'], axis=1)
combine = [train_df, test_df]

train_df.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked	Title
0	0	3	0	1	1	0	7.2500	S	1
1	1	1	1	2	1	0	71.2833	C	3
2	1	3	1	1	0	0	7.9250	S	2
3	1	1	1	2	1	0	53.1000	S	3
4	0	3	0	2	0	0	8.0500	S	1

泰坦尼克号数据分析流程

（6）整理、清洗数据

结合SibSp和Parch特征创建一个新特征FamilySize，意为包括兄弟姐妹、配偶、父母、孩子和自己的所有家人数量

```
for dataset in combine:
    dataset['FamilySize'] = dataset['SibSp'] + dataset['Parch'] + 1

train_df[['FamilySize', 'Survived']].groupby(['FamilySize'],as_index=False).mean().
sort_values(by='Survived', ascending=False)
```

	FamilySize	Survived
3	4	0.724138
2	3	0.578431
1	2	0.552795
6	7	0.333333
0	1	0.303538
4	5	0.200000
5	6	0.136364
7	8	0.000000
8	11	0.000000

泰坦尼克号数据分析流程

（6）清洗数据、特征工程

创建一个新特征IsAlone，取值为0表示不是独自一人，取值为1表示独自一人

```
# 创建新特征IsAlone
for dataset in combine:
        dataset['IsAlone'] = 0
        dataset.loc[dataset['FamilySize'] == 1, 'IsAlone'] = 1

train_df[['IsAlone', 'Survived']].groupby(['IsAlone'], as_index=False).mean()
```

	IsAlone	Survived
0	0	0.505650
1	1	0.303538

泰坦尼克号数据分析流程

（6）清洗数据、特征工程

舍弃 Parch、SibSp 和 FamilySize 特征，转而支持 IsAlone，因为 IsAlone 更能反映其与Survived的相关性

```
train_df = train_df.drop(['Parch', 'SibSp', 'FamilySize'], axis=1)
test_df = test_df.drop(['Parch', 'SibSp', 'FamilySize'], axis=1)
combine = [train_df, test_df]

train_df.head()
```

	Survived	Pclass	Sex	Age	Fare	Embarked	Title	IsAlone
0	0	3	0	1	7.2500	S	1	0
1	1	1	1	2	71.2833	C	3	0
2	1	3	1	1	7.9250	S	2	1
3	1	1	1	2	53.1000	S	3	0
4	0	3	0	2	8.0500	S	1	1

泰坦尼克号数据分析流程

（6）清洗数据、特征工程

填补分类特征Embarked

登船港口特征Embarked，有三种可能取值 S、Q、C。仅训练数据集有两个缺失值，采用众数填补缺失值。

```
freq_port = train_df.Embarked.dropna().mode()[0]
for dataset in combine:
    dataset['Embarked'] = dataset['Embarked'].fillna(freq_port)

train_df[['Embarked',
'Survived']].groupby(['Embarked'],as_index=False).mean().sort_values(by='Survived',
ascending=False)
```

	Embarked	Survived
0	C	0.553571
1	Q	0.389610
2	S	0.339009

泰坦尼克号数据分析流程

（6）清洗数据、特征工程

转换分类特征Embarked为序数

```
for dataset in combine:
      print(dataset['Embarked'].isnull().sum())

# 找到 Embarked 列的众数（出现最频繁的值）
most_common_embarked = train_df['Embarked'].mode()[0]  # 假设 train_df 是 combine 中的一个数据集

# 填充缺失值
for dataset in combine:
   dataset['Embarked'] = dataset['Embarked'].fillna(most_common_embarked)

for dataset in combine:
   print(dataset['Embarked'].unique())

#for dataset in combine:
   #dataset['Embarked'] = dataset['Embarked'].map( {'S': 0, 'C': 1, 'Q': 2} ).astype(int)
train_df.head()
```

	Survived	Pclass	Sex	Age	Fare	Embarked	Title	IsAlone	Age*Pclass
0	0	3	0	1	7.2500	0	1	0	3
1	1	1	1	2	71.2833	1	3	0	2
2	1	3	1	1	7.9250	0	2	1	3
3	1	1	1	2	53.1000	0	3	0	2
4	0	3	0	2	8.0500	0	1	1	6

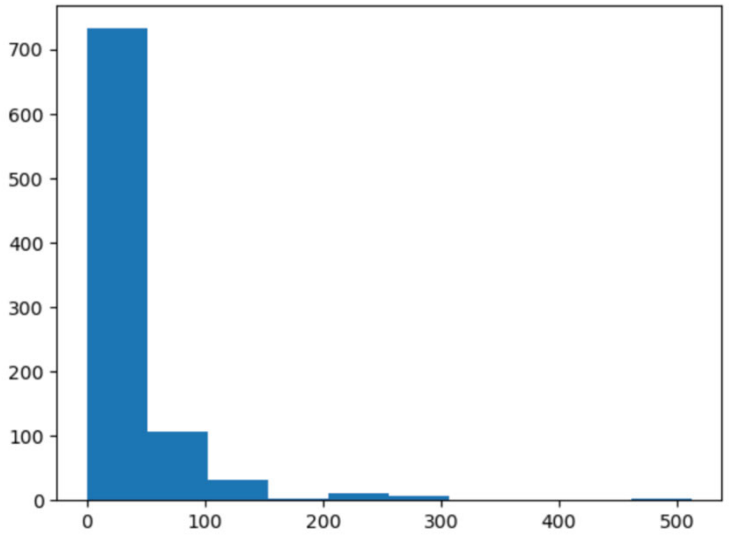
泰坦尼克号数据分析流程

（6）清洗数据、特征工程

填补票价Fare缺失值并可视化

```
# 测试集中Fare有一个缺失值，用中位数进行填补
test_df['Fare'].fillna(test_df['Fare'].dropna().median(), inplace=True)

plt.hist(train_df['Fare'])
```



泰坦尼克号数据分析流程

（6）清洗数据、特征工程

对票价Fare进行分箱

```
# 根据样本分位数进行分箱，等频分箱
train_df['FareBand'] = pd.qcut(train_df['Fare'] , 4)
train_df[['FareBand',
'Survived']].groupby(['FareBand'],as_index=False).mean().sort_values(by='FareBand', ascending=True)
```

	FareBand	Survived
0	(-0.001, 7.91]	0.197309
1	(7.91, 14.454]	0.303571
2	(14.454, 31.0]	0.454955
3	(31.0, 512.329]	0.581081

泰坦尼克号数据分析流程

（6）清洗数据、特征工程

将票价Fare替换为序数

```
for dataset in combine:
        dataset.loc[ dataset['Fare'] <= 7.91, 'Fare'] = 0
        dataset.loc[(dataset['Fare'] > 7.91) & (dataset['Fare'] <= 14.454), 'Fare'] = 1
        dataset.loc[(dataset['Fare'] > 14.454) & (dataset['Fare'] <= 31), 'Fare'] = 2
        dataset.loc[ dataset['Fare'] > 31, 'Fare'] = 3
        dataset['Fare'] = dataset['Fare'].astype(int)

train_df = train_df.drop(['FareBand'], axis=1)
combine = [train_df, test_df]

train_df.head(5)
```

	Survived	Pclass	Sex	Age	Fare	Embarked	Title	IsAlone	Age*Pclass
0	0	3	0	1	0	0	1	0	3
1	1	1	1	2	3	1	3	0	2
2	1	3	1	1	1	0	2	1	3
3	1	1	1	2	3	0	3	0	2
4	0	3	0	2	1	0	1	1	6

泰坦尼克号数据分析流程

（7）构建模型并预测结果

我们的问题是想确定输出（幸存与否）与其他变量或特征（性别、年龄、票价等级...）之间的关系，这属于典型的分类和回归问题。

当使用给定的数据集训练我们的模型时，我们称为监督式学习的机器学习。这样，模型主要有：

- 逻辑回归
- 支持向量机
- KNN 或 k-最近邻
- 朴素贝叶斯分类器
- 决策树
- 随机森林
- 感知机
- 人工神经网络
- RVM 或相关向量机

泰坦尼克号数据分析流程

（7）构建模型并预测结果

确定训练集特征 X_train、标签 Y_train 和测试集特征 X_test

```
X_train = train_df.drop("Survived", axis=1)
Y_train = train_df["Survived"]
X_test = test_df.drop("PassengerId", axis=1).copy()

X_train.shape, Y_train.shape, X_test.shape
```

```
((891, 8), (891,), (418, 8))
```

泰坦尼克号数据分析流程

（7）构建模型并预测结果

逻辑回归

逻辑回归是以线性回归为理论支持，但又通过sigmoid函数（逻辑回归函数）引入非线性因素，用来测量分类因变量和一个或多个自变量关系的模型，最常见的就是用来处理二分类问题。我们关注模型基于训练集生成的置信度分数。

```
# 逻辑回归模型
drop_columns = ['PassengerId', 'Name', 'Ticket', 'Cabin']
X_train = X_train.drop(drop_columns, axis=1)
drop_columns = [ 'Name', 'Ticket', 'Cabin']
X_test = X_test.drop(drop_columns, axis=1)
logreg = LogisticRegression()
logreg.fit(X_train, Y_train)
Y_pred = logreg.predict(X_test) # logreg.predict_proba(X_test)[:,1]
acc_log = round(logreg.score(X_train, Y_train) * 100, 2)
print(f'逻辑回归模型准确率：{acc_log}%')
```

逻辑回归模型准确率：79.24%

泰坦尼克号数据分析流程

（7）构建模型并预测结果

逻辑回归

可以使用逻辑回归中特征的系数，来验证我们对特征创建和完成目标的假设正确与否。
正系数会增加响应的对数几率（从而增加概率），而负系数会降低响应的对数几率（从而降低概率）。

```
coeff_df = pd.DataFrame(train_df.columns.delete(0))
coeff_df.columns = ['Feature']
coeff_df["Correlation"] = pd.Series(logreg.coef_[0])
coeff_df.sort_values(by='Correlation', ascending=False)
```

- Sex是最高的正系数，意味着随着性别值的增加（男性：0 到女性：1），Survived=1 的概率增加越多；
- Title是第二高的正相关特征；
- 相反，随着 Pclass 的增加，Survived=1 的概率降低越多；
- Age是第二高的负相关特征，即随着年龄的增加，Survived=1 的概率降低越多，幸存概率越小；
- Age * Pclass 对应系数绝对值较小，可能不是一个很好的人工特征。

泰坦尼克号数据分析流程

（7）构建模型并预测结果

支持向量机

支持向量机是一类按监督学习方式对数据进行二元分类的广义线性分类器，它的决策边界是对学习样本求解的最大边距超平面。最为常见的就是通过核函数的方法进行非线性分类。可以看到，该模型生成的置信度得分高于逻辑回归模型。

```
# 支持向量机模型
svc = SVC()
svc.fit(X_train, Y_train)
Y_pred = svc.predict(X_test)
acc_svc = round(svc.score(X_train, Y_train) * 100, 2)
print(f'支持向量机模型准确率：{acc_svc}%')
```

支持向量机模型准确率：82.04%

泰坦尼克号数据分析流程

（7）构建模型并预测结果

朴素贝叶斯分类器

朴素贝叶斯分类器是一系列以假设特征之间强独立下运用贝叶斯定理为基础的简单概率分类器。模型生成的置信度得分是目前评估的模型中最低的。

```
# 朴素贝叶斯分类器
gaussian = GaussianNB()
gaussian.fit(X_train, Y_train)
Y_pred = gaussian.predict(X_test)
acc_gaussian = round(gaussian.score(X_train, Y_train) * 100, 2)
print(f'朴素贝叶斯分类器模型准确率：{acc_gaussian}%')
```

朴素贝叶斯分类器模型准确率：77.44%

泰坦尼克号数据分析流程

（7）构建模型并预测结果

感知机

感知机是一种用于监督学习二元分类器的算法（可以决定由数字向量表示的输入是否属于某个特定类的函数）。 它是一种线性分类器，即基于将一组权重与特征向量相结合的线性预测函数进行预测的分类算法

```
# 感知机
perceptron = Perceptron()
perceptron.fit(X_train, Y_train)
Y_pred = perceptron.predict(X_test)
acc_perceptron = round(perceptron.score(X_train, Y_train) * 100, 2)
print(f'感知机模型准确率：{acc_perceptron}%')
```

感知机模型准确率：71.94%

泰坦尼克号数据分析流程

（7）构建模型并预测结果

线性SVC

线性SVC是一种用于分类问题的机器学习算法，其核心思想是寻找一个线性决策边界（即超平面），将不同类别的样本尽可能清晰地分开。

```
# 朴素贝叶斯分类器
gaussian = GaussianNB()
gaussian.fit(X_train, Y_train)
Y_pred = gaussian.predict(X_test)
acc_gaussian = round(gaussian.score(X_train, Y_train) * 100, 2)
print(f'朴素贝叶斯分类器模型准确率：{acc_gaussian}%')
```

朴素贝叶斯分类器模型准确率：77.44%

泰坦尼克号数据分析流程

（7）构建模型并预测结果

SDG

SDG分类器基于随机梯度下降的线性分类模型，即它假设数据点满足线性可分或近似线性可分的情况。它的决策边界是一个超平面，具体的分类方式取决于所选的损失函数。

```
# SDG随机梯度下降
sgd = SGDClassifier()
sgd.fit(X_train, Y_train)
Y_pred = sgd.predict(X_test)
acc_sgd = round(sgd.score(X_train, Y_train) * 100, 2)
print(f'SDG模型准确率：{acc_sgd}%')
```

SDG模型准确率：65.88%

泰坦尼克号数据分析流程

（7）构建模型并预测结果

决策树

决策树是将特征（树枝）映射到目标值（树叶）的分类或回归方法。目标变量可以取一组有限值的树模型称为分类树；在这些树结构中，叶子代表类标签，分支代表导致这些类标签的特征的结合。目标变量可以取连续值的决策树称为回归树。模型置信度得分是目前最高的。

```
# 决策树
decision_tree = DecisionTreeClassifier()
decision_tree.fit(X_train, Y_train)
Y_pred = decision_tree.predict(X_test)
acc_decision_tree = round(decision_tree.score(X_train, Y_train) * 100, 2)
print(f'决策树模型准确率：{acc_decision_tree}%')
```

决策树模型准确率：85.63%

泰坦尼克号数据分析流程

（7）构建模型并预测结果

随机森林

随机森林是一种用于分类、回归和其他任务的集成学习方法。它通过自助法（bootstrap）重采样技术，从原始训练样本集中有放回地重复随机抽取n个样本生成新的训练样本集合训练决策树，然后按以上步骤生成m棵决策树组成随机森林，新数据的分类结果按分类树投票多少形成的分数而定。其实质是对决策树算法的一种改进，将多个决策树合并在一起，每棵树的建立依赖于独立抽取的样本。模型置信度得分是目前评估的模型中最高的。

```
# 随机森林
random_forest = RandomForestClassifier(n_estimators=100)
random_forest.fit(X_train, Y_train)
Y_pred = random_forest.predict(X_test)
random_forest.score(X_train, Y_train)
acc_random_forest = round(random_forest.score(X_train, Y_train) * 100, 2)
print(f'随机森林模型准确率：{acc_random_forest}%')
```

随机森林模型准确率：85.63%

泰坦尼克号数据分析流程

（8）模型评估

我们现在可以对所有模型评估结果进行排名，以选择最适合我们问题的模型。 虽然决策树和随机森林的得分相同，但我们选择使用随机森林，因为它纠正了决策树过度拟合训练集带来的缺陷。

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier()
knn.fit(X_train, Y_train)
acc_knn = round(knn.score(X_train, Y_train) * 100, 2)

models = pd.DataFrame({
'Model': ['Support Vector Machines', 'KNN', 'Logistic Regression',
'Random Forest',
'Naive Bayes', 'Perceptron', 'Stochastic Gradient Decent', 'Linear
SVC', 'Decision Tree'],
'Score': [acc_svc, acc_knn, acc_log, acc_random_forest,
acc_gaussian, acc_perceptron, acc_sgd, acc_svc,
acc_decision_tree]})
models.sort_values(by='Score', ascending=False)
```

	Model	Score
3	Random Forest	85.63
8	Decision Tree	85.63
1	KNN	83.73
0	Support Vector Machines	82.04
7	Linear SVC	82.04
2	Logistic Regression	79.24
4	Naive Bayes	77.44
5	Perceptron	71.94
6	Stochastic Gradient Decent	71.94

Q & A