

# 棕榈学院

## 7 天 Python 进阶训练营讲义

### 第一讲



2019 年 1 月 14 日-2019 年 1 月 20 日

# 目录

## 项目介绍

### 一、定义并理解问题

### 二、获取数据

### 三、观察并理解数据

### 四、作业

## 结语



## 项目介绍

### （一）课程目标

本次训练营课程，内容设置清晰明了，主要目的是通过 Kaggle 数据实战，了解数据分析的代码全貌，并初步运用机器学习。

### （二）Kaggle 介绍

Kaggle 于 2010 年在墨尔本被创立，主要为开发商和数据科学家们提供举办机器学习竞赛、托管数据库、编写和分享代码的平台，是知名的提供数据、进行竞赛的网站。

其最重要的价值，体现在竞赛者可以尽可能地去使用和设计建模方法以解决现实中的难题，促进社会的发展。

而 Titanic 是其中最基础的数据之一，可以对其进行机器学习和数据分析，所以选取这样一个 project 让大家可以走进数据科学，开始学习 Python。

### （三）案例介绍

#### **Titanic: Machine Learning from Disaster**

泰坦尼克号的沉没是历史上伤亡人数最多、影响最深远的沉船事件之一。1921 年 4 月 15 日，泰坦尼克号与冰山相撞，2224 名乘客和船员中有 1502 人丧生，这场轰动性的悲剧震惊了国际社会，但也因此产生了更好的船舶安全法规。这次海难造成人员伤亡的原因之一是没有足够的救生船供乘客和船员使用。

虽然在沉船中幸存下来有一些运气因素，但有些人比其他人更可能存活下来，在本次学习中，在给定的部分存活人员信息中，将应用 Python 来分析哪些乘客可能在悲剧中存活。此次，棕榈学院将携手 Yiya 导师给各位想学习 Python，想要在数据行业继续发展学习的同学来讲授如何完成这样一个 project，相信会对你们的数据分析技能的提升大有裨益。

学院君提醒🔔: <https://www.kaggle.com/c/titanic>

这里为大家附上 Kaggle 中 Titanic 案例的网站地址，Titanic 的简介、相关数据等课程中需要用到的内容在这里都可以找到哦！大家也可以多多探索 Kaggle 上的其他案例鸭~

## （四）课程安排

一般来说，一个机器学习的 project 分为 8 部分：

1. 定义并理解问题
2. 获取数据
3. 观察并理解数据
4. 数据清理&数据可视化
5. 数据需求&特征工程
6. 选取模型
7. 测试训练集&测试集
8. 衡量结果

在本次课程的学习中将逐步展开研究。



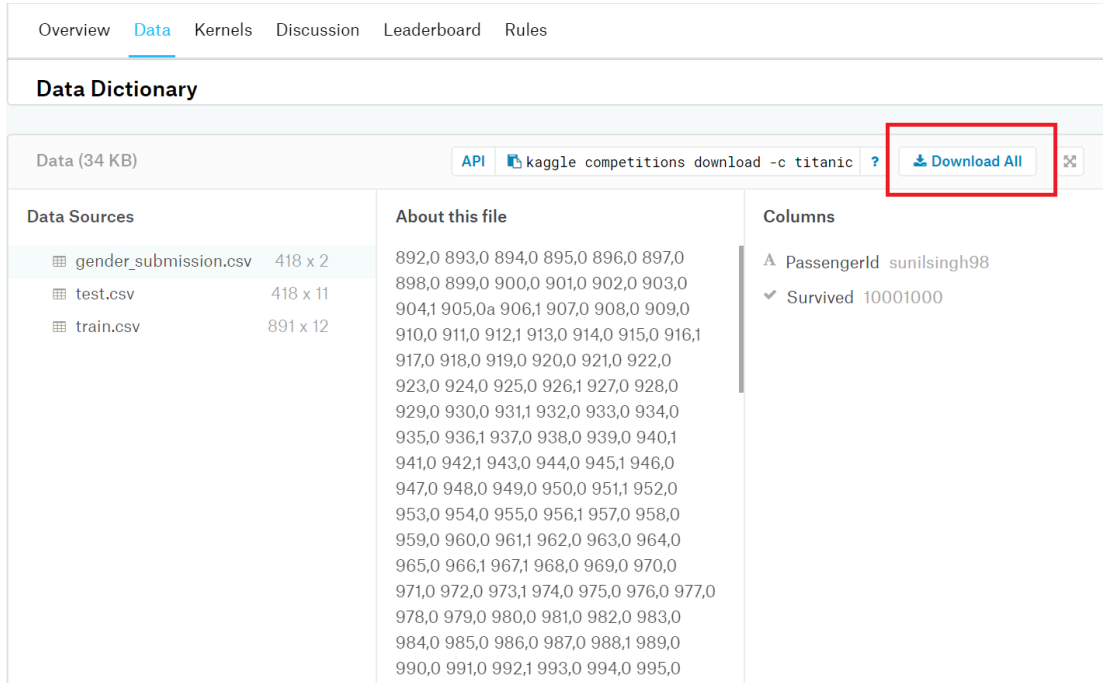
## 一、定义并理解问题。

因为每个问题所要解决的问题方向是不同的，所以在开展数据分析的时候，首先要定义并理解问题。在这个问题中的目的是要预测泰坦尼克号上的游客能否存活，结果只有两个：幸存或遇难。所以相当于是一个二元分类方法。最终要把结果分为两类：1. 存活，2. 死去。衡量的方法是用“accuracy”，即准确度，也就是说我们的预测有多少是准确的，但这只是一种衡量方法，其他的 project 会有其他衡量方法，这也是需要注意的。



## 二、获取数据

在这个 project 中比较简单，因为 Kaggle 网站已经为我们提供（Kaggle-Titanic——Data——Data Sources——test（测试集）和 train（训练集）），不需要自己去搜集其他数据，直接下载即可。



Overview **Data** Kernels Discussion Leaderboard Rules

**Data Dictionary**

Data (34 KB) [API](#) [kaggle competitions download -c titanic](#) [Download All](#)

Data Sources	About this file	Columns
<ul style="list-style-type: none"><li>gender_submission.csv 418 x 2</li><li>test.csv 418 x 11</li><li>train.csv 891 x 12</li></ul>	<pre>892,0 893,0 894,0 895,0 896,0 897,0 898,0 899,0 900,0 901,0 902,0 903,0 904,1 905,0a 906,1 907,0 908,0 909,0 910,0 911,0 912,1 913,0 914,0 915,0 916,1 917,0 918,0 919,0 920,0 921,0 922,0 923,0 924,0 925,0 926,1 927,0 928,0 929,0 930,0 931,1 932,0 933,0 934,0 935,0 936,1 937,0 938,0 939,0 940,1 941,0 942,1 943,0 944,0 945,1 946,0 947,0 948,0 949,0 950,0 951,1 952,0 953,0 954,0 955,0 956,1 957,0 958,0 959,0 960,0 961,1 962,0 963,0 964,0 965,0 966,1 967,1 968,0 969,0 970,0 971,0 972,0 973,1 974,0 975,0 976,0 977,0 978,0 979,0 980,0 981,0 982,0 983,0 984,0 985,0 986,0 987,0 988,1 989,0 990,0 991,0 992,1 993,0 994,0 995,0</pre>	<ul style="list-style-type: none"><li>PassengerId sunilsingh98</li><li>Survived 10001000</li></ul>

但是，在生活中自己做的 project 可能就需要自己去搜集数据，比如做爬虫等得到一些数据。大家也要注意培养搜集数据的能力。

## 三、观察并理解数据

学院君提醒🕒：下面的代码图片中#后的部分是学院君提醒大家的注意事项和步骤讲解，只需要看懂就可以了哦，在自己操作的时候不用添加~

### （一）准备步骤（\*非必需）

在这里 import warnings，把 warning 忽视掉，但这一步可需要可不需要。

```
In [25]: #有时候运行代码时会有很多warning输出，如提醒新版本之类的，如果不想这些乱糟糟的输出可以这样；但是切记，不要盲目设置取消输出。  
import warnings  
warnings.filterwarnings('ignore')
```

### （二）导入数据

然后先开始导入数据，用的是最常见的 pandas 数据包。因为数据是 CSV 文件，所以直接把 pandas 导入后，直接用 pd.read\_csv 和 pd.read\_csv 来导入数据训练集和测试集。

注意！两个文件要放在正确的文件夹内。文件夹名称（即下图中标红的部分）使用单引号双引号均可。

```
In [4]: #载入pandas数据包，以pd来简化命名；Pandas是Python一个非常重要的科学计算包，数据科学家、数据分析师都会非常普遍地用到这个包；  
import pandas as pd
```

```
In [1]: #读取准备好的CSV数据，导入训练集和测试集  
train_df = pd.read_csv('train.csv')  
test_df = pd.read_csv('test.csv')
```

【这里学院君再给大家复习一下在 Python 入门课程中我们学到的数据导入的方法】

1. 使用 open 读取文件

```
file = open('文件路径', 'r')  
print(file.read())
```

2. 使用 pandas 来读取文件

例：

首先载入 pandas，代码为 import pandas as pd

（如果在前面引进了就不需要了）

```
df = pd.read_csv('data.csv', index_col = 0)
```

df

### (1) 读取 excel 文件

例:

```
df = pd.read_excel('data.xlsx')
```

df

### (2) 读取 txt 文件

例:

```
df = pd.read_table('data.txt', sep = ',', index_col = 0) df
```

### (3) 如何存储文件

例:

```
df.to_excel('dat.xlsx')
```

### (4) 读取复杂格式的 txt

例:

我们只想读取数据，不想要废话

```
df = pd.read_table('data1.txt', sep = ',', skiprows = [0, 1, 2, 3], header = None,
                  names = ['n1', 'n2', 'n3', 'n4', 'message'],
                  index_col = ['message'], nrows = 5)
```

提示:

[https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read\\_table.html](https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_table.html)

这个网页里介绍了 `pandas.read_table` 如果大家遇到不会的函数时，可以去找这些函数的说明文档查看。

注：这里主要是帮助大家复习以前的知识，如果数据不存在的话，直接复制代码是无法执行的，请同学们根据自己想要读取的数据修改代码。

之后可以尝试查看训练集和测试集的数据维度：

```
In [13]: #查看训练集和测试集的数据维度
         train_df.shape, test_df.shape

Out[13]: ((891, 12), (418, 11))
```

可以看到训练集有 891 行，12 列；测试集有 418 行，11 列。

**【思考】**为什么测试集会少一列呢？

因为测试集中并没有告诉最终结果，这正是我们在测试集中需要预测的。



## 【这里学院君科普一下训练集和测试集的区别】

**训练集：**学习样本数据集，通过匹配一些参数来建立一个分类器。建立一种分类的方式，主要是用来训练模型的。

**测试集：**主要是测试训练好的模型的分辨能力（识别率等）。

简单来说呢，训练集和测试集就是“五年高考三年模拟”的区分。平时做题都在用训练集，考试的时候就要用测试集来测试模型的真实水平了。这样解释是不是好理解很多了呢~

## （三）观察并理解数据

### 1. 载入 matplotlib 包并读取数据

```
In [22]: #载入matplotlib包，以plt简化命名；Matplotlib是Python中一个非常重要的绘图包，大量的数据可视化工作会用到这个包；  
import matplotlib.pyplot as plt
```

```
In [16]: #读取训练集表头-就是每一个表格的Column name，列名  
print(train_df.columns.values)  
  
['PassengerId' 'Survived' 'Pclass' 'Name' 'Sex' 'Age' 'SibSp' 'Parch'  
 'Ticket' 'Fare' 'Cabin' 'Embarked']
```

```
In [17]: #读取测试集表头-同上  
print(test_df.columns.values)  
  
['PassengerId' 'Pclass' 'Name' 'Sex' 'Age' 'SibSp' 'Parch' 'Ticket' 'Fare'  
 'Cabin' 'Embarked']
```

```
In [14]: #观望一下数据，这个函数可以供你看到了表头的前五行数据  
train_df.head()
```

```
Out[14]:
```

	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

变量分为两类：

（1）**分类变量：**比如 PassengerId, Pclass, Name, Sex 等。这些变量并不是连续性的，都是以不同分类区分开的。

（2）**数字变量：**比如 Age, Fare 等，最终可能要直接影响数据分析。

### 2. 用 info function 简单看一下每一列的格式：整形、对象、小数

```
In [18]: #这个函数可以查看所有数据变量的类型信息, int=integer(整数), object=对象, float=浮点数
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 PassengerId    891 non-null int64
  Survived      891 non-null int64
  Pclass        891 non-null int64
  Name          891 non-null object
  Sex           891 non-null object
  Age           714 non-null float64
  SibSp         891 non-null int64
  Parch        891 non-null int64
  Ticket       891 non-null object
  Fare         891 non-null float64
  Cabin        204 non-null object
  Embarked     889 non-null object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.6+ KB
```

在这里也可以看到有些数据是缺失的, 比如 Age 其实只有 714, 而根据数据维度以可看到本应该有 819 个, 那么有 100 多个是缺失的。

也可以用另一个 function, 看是不是有缺失值, 可以得到每一列缺失的数据。

```
In [69]: #判断数据里每一个数据变量的缺失值有多少个, 我们可以看到Age这个变量的缺失值是最多的;
train_df.isnull().sum()
```

```
Out[69]: 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
```

【拓展】缺失的数据该怎么办呢?

对于缺失的数据的处理方法会在接下里的课程中详细讲述~

### 3. 用 describe function 得到数据基本知识

```
In [19]: #对数据进行描述性统计分析, 这个函数可以自动计算出每一个数字变量的平均数、标准差、分位数等重要统计概念
train_df.describe()
```

```
Out[19]:
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

但这里只会对数字的变量进行计算等操作, 因为需要计算均值方差、最大值、最小值等, 所以比如名字性别这些无法进行计算。

学院君提醒🔔：导出表格的每一列标题在 Kaggle 的 Titanic 项目有对应的具体介绍。

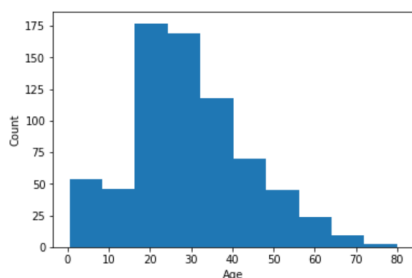
Overview	Data	Kernels	Discussion	Leaderboard	Rules
Data Dictionary					
Variable	Definition	Key			
survival	Survival	0 = No, 1 = Yes			
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd			
sex	Sex				
Age	Age in years				
sibsp	# of siblings / spouses aboard the Titanic				
parch	# of parents / children aboard the Titanic				

## 4. 做一些图表来进一步了解数据

(1) 用 histogram 对年龄进行绘图，可以看看在这艘游轮上的乘客年龄分布

```
In [27]: #绘制年龄分布直方图，横轴为年龄，纵轴为人数
plt.hist(train_df['Age'])
plt.xlabel('Age')
plt.ylabel('Count')
```

Out[27]: Text(0, 0.5, 'Count')



Histogram 是其中一种表格，barplot 等也可以运用。

可以看到：基本 20-40 岁年轻人最多，小孩子和老人各占一小部分。

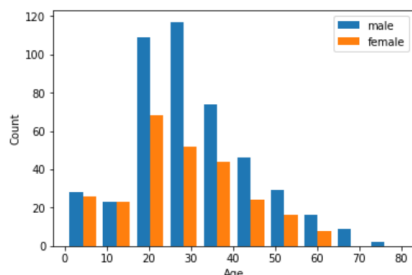
学院君提醒🔔：

- ✓ plt 是前面导入的 matplotlib 的数据包。
- ✓ train\_df 是整个训练集，用中括号内引号打入名字可以直接得到一系列的数据。
- ✓ 键入 label, x 轴年龄, y 轴数量。

## (2) 区分性别绘图

```
In [43]: #这是一个绘图函数，从matplotlib这个包里调用，绘制各年龄层性别分布直方图
plt.hist([train_df[train_df['Sex'] == 'male'].Age, train_df[train_df['Sex'] == 'female'].Age])
plt.legend(['male', 'female'])
plt.xlabel('Age')
plt.ylabel('Count')
```

Out[43]: Text(0, 0.5, 'Count')



Sex 只有两个可能：male female，绘图时候相当于有两列数据，所以要加中括号把这两列数据用逗号分开，接下来再进行绘图。

可以看到：20-30 岁的男性乘客比女性多很多。

学院君提醒🔔：

.age 和 ['age'] 一样，都是得到一列数据的方式。

## (3) 用 crosstab 的 function 生成存活与死亡人数中性别分布交叉表

```
In [44]: #生成交叉表，分析存活与死亡人数中性别分布
pd.crosstab(train_df['Survived'], train_df['Sex'])
```

Out[44]:

Sex	female	male
Survived		
0	81	468
1	233	109

用 crosstab 的 function，相当于把数据幸存与否的列和性别的列提取出来生成 2x2 小表格。

可以看到：女性大概有 300 多位，存活率相对较高，有 200 多位。男性一共 200 多人，只有 100 多人存活下来。这可能和男性让女性和老人小孩先上救生船有着很大关系。所以在预测时候可知性别为男性会是影响生存率的一个重要因素。

## (4) 用 crosstab 的 function 生成客舱等级和生存率关系交叉表

```
In [59]: #生成交叉表，分析存活与死亡人数中客舱等级分布
pd.crosstab(train_df['Survived'], train_df['Pclass'])
```

Out[59]:

Pclass	1	2	3
Survived			
0	80	97	372
1	136	87	119

学院君提醒🔔:

客舱等级的 1、2、3 的具体含义如果不太清楚，记得回到 Titanic 资料页查看 Data Dictionary 哦~

Overview [Data](#) Kernels Discussion Leaderboard Rules

Data Dictionary

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd

根据 Data Dictionary 得知，1 为一等舱，等级最高，以此类推适用。

可以看到：一等舱两百多个人有百分之六十的人存活下来，三等舱存活率几乎只有百分之二十五。可以设想到一等舱房间较大，设备完善，逃生时候更加顺利等，所以客舱等级也可能是一个重要因素。

#### (5) 在训练集中用 groupby 的 function 查看登陆港口和幸存的情况

```
In [60]: #根据登陆城市、存活情况对数据进行分组，并统计个数
train_df.groupby(['Embarked', 'Survived']).size()

Out[60]: Embarked  Survived
C              0           75
           1           93
Q              0           47
           1           30
S              0          427
           1          217
dtype: int64
```

在这里不仅可以用 size，size、max、min 等都可以计算。

可以看到，C 港口登陆有近 50%的人存活，而 S 港口仅有 30%最后存活下来。可以推测这也会是要用到的很重要的因素。

#### (6) 除了这几种运用到的对数据进行了解的情况，也可以用到 sql，通过 sql 统计存活情况、乘客同船的父/子女人数

```
In [51]: #这个函数其实是SQL语言在Python里的一个应用，SQL语言主要负责数据管理，这句代码的意思是对python中的pandas进行sql查询
#为了方便使用，我们不用下载SQL语言，而是直接在Python里使用
from pandasql import sqldf

In [62]: #从训练集中选择存活情况、乘客同船的父/子女人数这几个变量进行分组计数（注意：其实这几句代码已经是SQL语言了哦）
#Select, from, Groupby这三个函数可是SQL语言里非常重要且应用普遍的函数了哦！
q1 = """
SELECT Survived, Parch, count(*)
FROM train_df
GROUP BY Survived, Parch
"""
sqldf(q1)
```

```
Out[62]:
```

	Survived	Parch	count(*)
0	0	0	445
1	0	1	53
2	0	2	40
3	0	3	2
4	0	4	4
5	0	5	4
6	0	6	1
7	1	0	233
8	1	1	65
9	1	2	40
10	1	3	3
11	1	5	1

注意！这些包安装之后并不一定会在电脑中，如果没有安装的话重新输入代码进行安装。

可以得到：没有父母孩子幸存的比有一个父母孩子的存活率会更高。可以推测也是一个重要因素。

学院君拔高笔记：

大家自己可以更多看数据、分析数据进行对比，并尝试设想是什么导致数据的变化，搜集出更多因素。

## 四、作业

根据第一节课上老师讲授的内容，对数据进行同样的处理，展示自己的代码和学习心得。

在小程序上截图打卡自己的学习成果～



## 结语

以上是对数据进行简单的可视化过程。接下来会进入到数据清理的学习。在数据中会有一些缺失的词，该如何进行处理这些问题呢？还有一些名字等比较长没有规则的变量该如何处理呢？下节课 Yiya 导师将为大家详细展开讲述。

学院君暖心小提示(♥ ω ♥)

如果感觉这节课困难比较大或是并没有上过 Python 入门课程的学员小伙伴，可以去找学院小动物们咨询入门课程哦~本课程的主题主要是通过 project 实战，进一步巩固 Python 课程的学，所以安装等基础问题不在此赘述，想要跟着老师的步伐进一步学习 Python 的小伙伴加油补课，大家一起进步鸭！



扫码关注棕榈学院，解锁更多精彩课程