Beijing Forest Studio

北京理工大学信息系统及安全对抗实验中心

数据挖掘项目实战

Titanic: Machine Learning from Disaster

郝靖伟　硕士研究生

2018年05月01日

内容提要

- 背景简介
- 总体思路
- 实战演练
- 参考文献

背景简介

什么是kaggle?

Google:

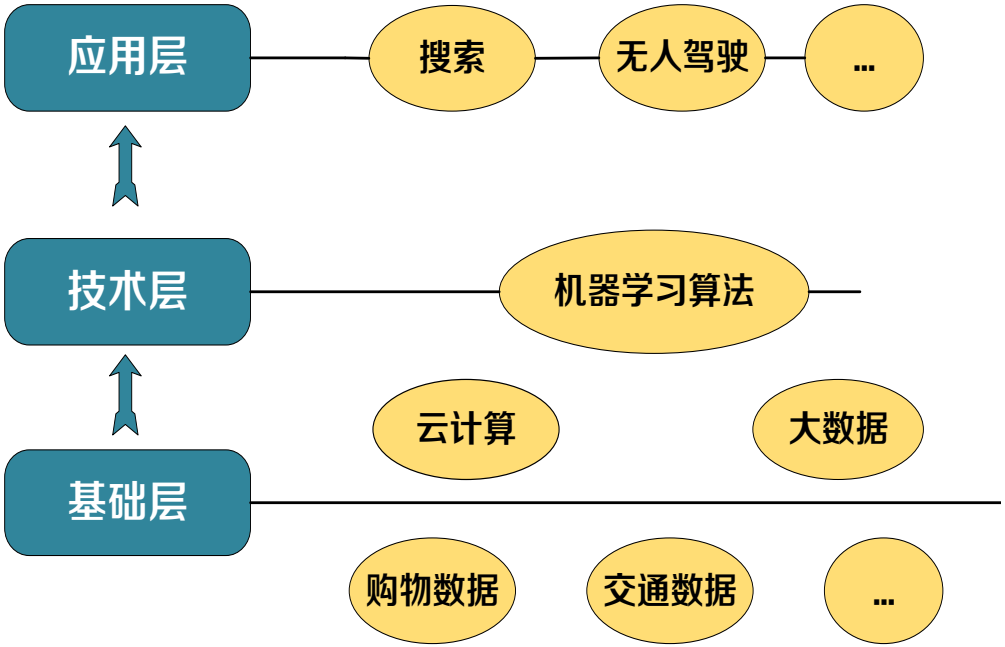
Kaggle is a platform for predictive **modelling and analytics competitions** in which statisticians and data miners compete to produce the best models for predicting and describing the datasets uploaded by companies and users.

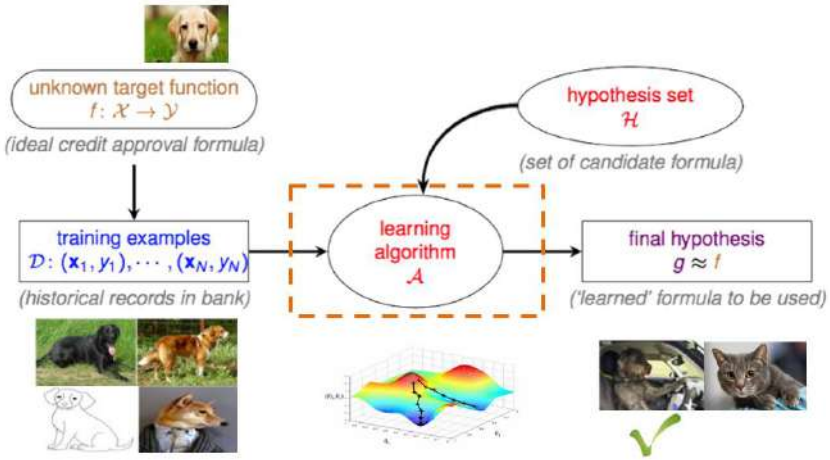
- 1.Find a competition
  - For learning
  - For prizes and points
- 2.Build your model
- 3.Submit your entry

背景介绍

- Kaggle网址：https://www.kaggle.com
- 入门比赛：
  - Getting Started
  - Playground
- 比赛类型：
  - 传统Business
  - 图像识别AI
- 相关教程：
  - Code: R, Python
  - Data analysis and visualization
  - Statistics

背景介绍


应用层
搜索
无人驾驶
...
技术层
机器学习算法
基础层
云计算
大数据
购物数据
交通数据
...


unknown target function
$f: \mathcal{X} \to \mathcal{Y}$
(ideal credit approval formula)
hypothesis set
$\mathcal{H}$
(set of candidate formula)
training examples
$\mathcal{D}: (\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$
(historical records in bank)
learning
algorithm
$\mathcal{A}$
final hypothesis
$g \approx f$
('learned' formula to be used)

总体思路

总体思路

提出问题	理解数据	数据清洗	构建模型	模型评估	方案实施
有监督学习问题，还是无监督问题？如果是有监督学习，那么是分类问题还是回归问题？ 如果是有监督学习，需要处理的标签数量有多少？维度有多少？ 存在数据不平衡问题吗？ …	对数据、需求或目标进行分析，进行一些必要的探索，如了解数据的大致结构、数据量、各特征的统计信息、整个数据质量情况、数据的分布情况等。为了更好体现数据分布情况，数据可视化是一个不错方法。	如存在缺失数据、数据不规范、数据分布不均衡、存在奇异数据、有很多非数值数据、存在很多无关或不重要的数据等等。	因此在实际选择时，一般会选用几种不同方法来训练模型，然后比较它们的性能，从中选择最优的这个。	如果我们对模型的测试结果满意，就可以用此模型对以后的进行预测；如果我们测试结果不满意，我们可以优化模型。	

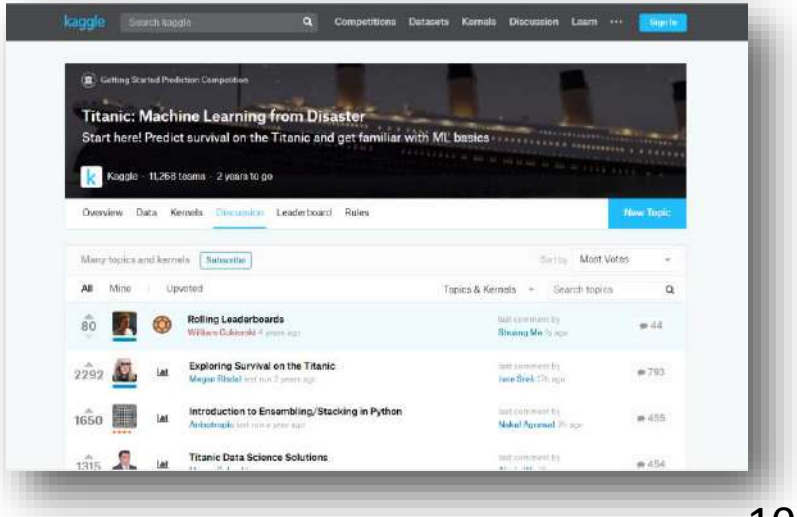
实战演练

提出问题

Titanic: Machine Learning from Disaster

我们研究的问题是：

什么样的人在泰坦尼克号中更容易存活?



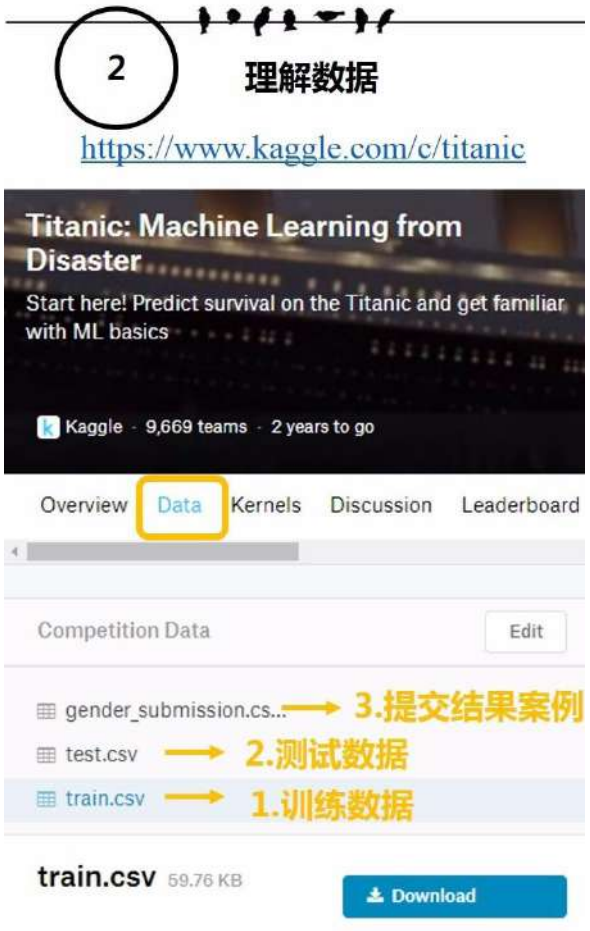
理解数据

数据不均衡问题

- 如何解决?
  - 让正负样本在训练过程中拥有相同的话语权
  - 采样：上采样（Oversampling）和下采样（Undersampling）
  - 数据合成方法：SMOTE、Borderline-SMOTE、ADASYN
  - 加权
  - 一分类（正负样本极不平衡的场景）：One-class SVM

- 如何选择?
  - 在正负样本都足够多且比例不是特别悬殊的情况下，考虑采样或者加权的方法
  - 在正负样本都非常之少的情况下，采用数据合成的方式
  - 在负样本足够多，正样本非常之少且比例及其悬殊的情况下，考虑一分类方法

理解数据



A	B	C	D	E	F	G	H	I	J	K	L
Passenger	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarke
1	0	3	Braund, M	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cumings,	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen	female	26	0	0	STON/O2	7.925		S
4	1	1	Futrelle, M	female	35	1	0	113803	53.1	C123	S
5	0	3	Allen, Mr.	male	35	0	0	373450	8.05		S
6	0	3	Moran, M	male		0	0	330877	8.4583		Q
7	0	1	McCarthy	male	54	0	0	17463	51.8625	E46	S
8	0	3	Palsson, M	male	2	3	1	349909	21.075		S
9	1	3	Johnson, I	female	27	0	2	347742	11.1333		S
10	1	2	Nasser, M	female	14	1	0	237736	30.0708		C
11	1	3	Sandstron	female	4	1	1	PP 9549	16.7	G6	S
12	1	1	Bonnell, M	female	58	0	0	113783	26.55	C103	S
13	0	3	Saunderc	male	20	0	0	A/5. 2151	8.05		S
14	0	3	Andersson	male	39	1	5	347082	31.275		S
15	0	3	Vestrom, I	female	14	0	0	350406	7.8542		S
16	1	2	Hewlett, M	female	55	0	0	248706	16		S
17	0	3	Rice, Mast	male	2	4	1	382652	29.125		Q
18	1	2	Williams, I	male		0	0	244373	13		S
19	0	3	Vander Pl	female	31	1	0	345763	18		S
20	1	3	Masselma	female		0	0	2649	7.225		C
21	0	2	Fynney, M	male	35	0	0	239865	26		S
22	1	2	Beesley, M	male	34	0	0	248698	13	D56	S
23	1	3	McGowan	female	15	0	0	330923	8.0292		Q
24	1	1	Sloper, Mi	male	28	0	0	113788	35.5	A6	S
25	0	3	Palsson, M	female	8	3	1	349909	21.075		S
26	1	3	Asplund, I	female	38	1	5	347077	31.3875		S
27	0	3	Emir, Mr.	male		0	0	2631	7.225		C
28	0	1	Fortune, M	male	19	3	2	19950	263	C23 C25 C	S
29	1	3	O'Dwyer,	female		0	0	330959	7.8792		Q

导入数据

❶
```
# 导入处理数据包
import numpy as np
import pandas as pd
```

❷
```
#训练数据集
train = pd.read_csv("./train.csv")
#测试数据集
test = pd.read_csv("./test.csv")
print ('训练数据集:',train.shape,'测试数据集:',test.shape)
```
训练数据集: (891, 12) 测试数据集: (418, 11)

❸
```
#合并数据集，方便同时对两个数据集进行清洗
full = train.append( test , ignore_index = True )
print ('合并后的数据集:',full.shape)
```

有监督、不存在数据不平衡问题

理解数据

查看数据集信息

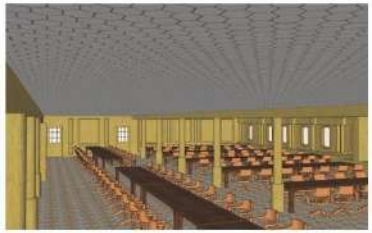
```
#查看数据
full.head()
```

	Age	Cabin	Embarked	Fare	Name
0	22.0	NaN	S	7.2500	Braund, Mr. Owen Harris

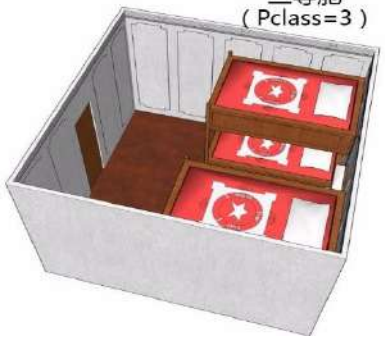
列名（变量）	含义
PassengerId	乘客编号
Survived	生存情况（1=存活，0=死亡）
Pclass	客舱等级（1=1等舱，2=2等舱，3=3等舱）
Name	姓名
Sex	性别
Age	年龄
SibSp	船上兄妹姐妹数/配偶数（同代直系亲属数）
Parch	船上父母数/子女数（不同代直系亲属数）
Ticket	船票编号
Fare	船票价格
Cabin	客舱号
Embarked	登船港口 出发地点：S=英国南安普顿 Southampton 途径地点1：C=法国 瑟堡市Cherbourg 出发地点2：Q=爱尔兰 昆士敦 Queenstown

头等舱接待室
（Pclass=1）

（Pclass=2）



三等舱
（Pclass=3）



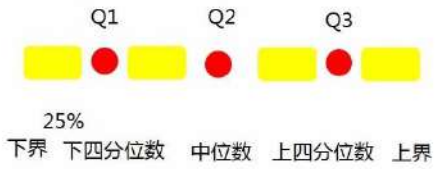
列名（变量）	含义	变量值
SibSp	船上兄妹姐妹数/配偶数（同代直系亲属数）	2
Parch	船上父母数/子女数（不同代直系亲属数）	1

理解数据

描述统计信息

```
#获取数据类型列的描述统计信息
full.describe()
```

		年龄 Age	船票价格 Fare	Parch
数据总数	count	1046.000000	1308.000000	1309.000000
平均值	mean	29.881138	33.295479	0.385027
标准差	std	14.413493	51.758668	0.865560
最小值	min	0.170000	0.000000	0.000000
下四分位数	25%	21.000000	7.895800	0.000000
中位数	50%	28.000000	14.454200	0.000000
上四分位数	75%	39.000000	31.275000	0.000000
最大值	max	80.000000	512.329200	9.000000



查看缺失数据

```
# 查看每一列的数据类型，和数据总数
full.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 12 columns):
年龄 Age            1046 non-null float64
船舱号 Cabin        295 non-null object
登船港口 Embarked   1307 non-null object
船票价格 Fare       1308 non-null float64
姓名 Name           1309 non-null object
不同代直系亲属 Parch 1309 non-null int64
乘客编号 PassengerId 1309 non-null int64
船舱等级 Pclass     1309 non-null int64
性别 Sex            1309 non-null object
同代直系亲属 SibSp  1309 non-null int64
生存情况 Survived   891 non-null float64
船票编码 Ticket     1309 non-null object
dtypes: float64(3), int64(4), object(5)
memory usage: 122.8+ KB
```

数据总数：行数

数据总数：列数

- 有4列数据有缺失值
- 船舱号（Cabin）里面数据总数是295，缺失了1309-295=1014，缺失率=1014/1309=77.5%
- 为后续数据清洗指明方向

数据清洗

数据预处理

1. 选择子集
2. 列名重命名
3. 缺失数据处理
4. 数据类型转换
5. 数据排序
6. 异常值处理

- **数据清洗**
  - 发现并纠正数据文件中可识别的错误的最后一道程序，包括检查数据一致性，处理无效值和缺失值等。

- 选择研究问题需要的数据
- 方便数据分析
- 缺失数据进一步处理
- 方便数据计算
- 发现更多有价值的信息
- 异常值处理使其符合定义范围

数据清洗

- 字符串类型缺失值处理
- 数据类型缺失值处理


```
#登船港口（Embarked）：查看里面数据长啥样
'''
出发地点：S=英国南安普顿Southampton
途径地点1：C=法国 瑟堡市Cherbourg
途径地点2：Q=爱尔兰 昆士敦Queenstown
'''
full['Embarked'].head()
```

```
0    S
1    C
2    S
3    S
4    S
Name: Embarked, dtype: object
```

```
'''
# 只有两个缺失值，我们将缺失值填充为最频繁出现的值：
S=英国南安普顿Southampton
'''
full['Embarked'] = full['Embarked'].fillna('S')
```

❶ 数据填充方法

```
#年龄(Age)
full['Age']=full['Age'].fillna( full['Age'].mean() )
#船票价格(Fare)
full['Fare'] = full['Fare'].fillna( full['Fare'].mean() )
```

❷ 用平均值填充

```
#船舱号（Cabin）：查看里面数据长啥样
full['Cabin'].head()
```

```
0      NaN
1      C85
2      NaN
3     C123
4      NaN
Name: Cabin, dtype: object
```

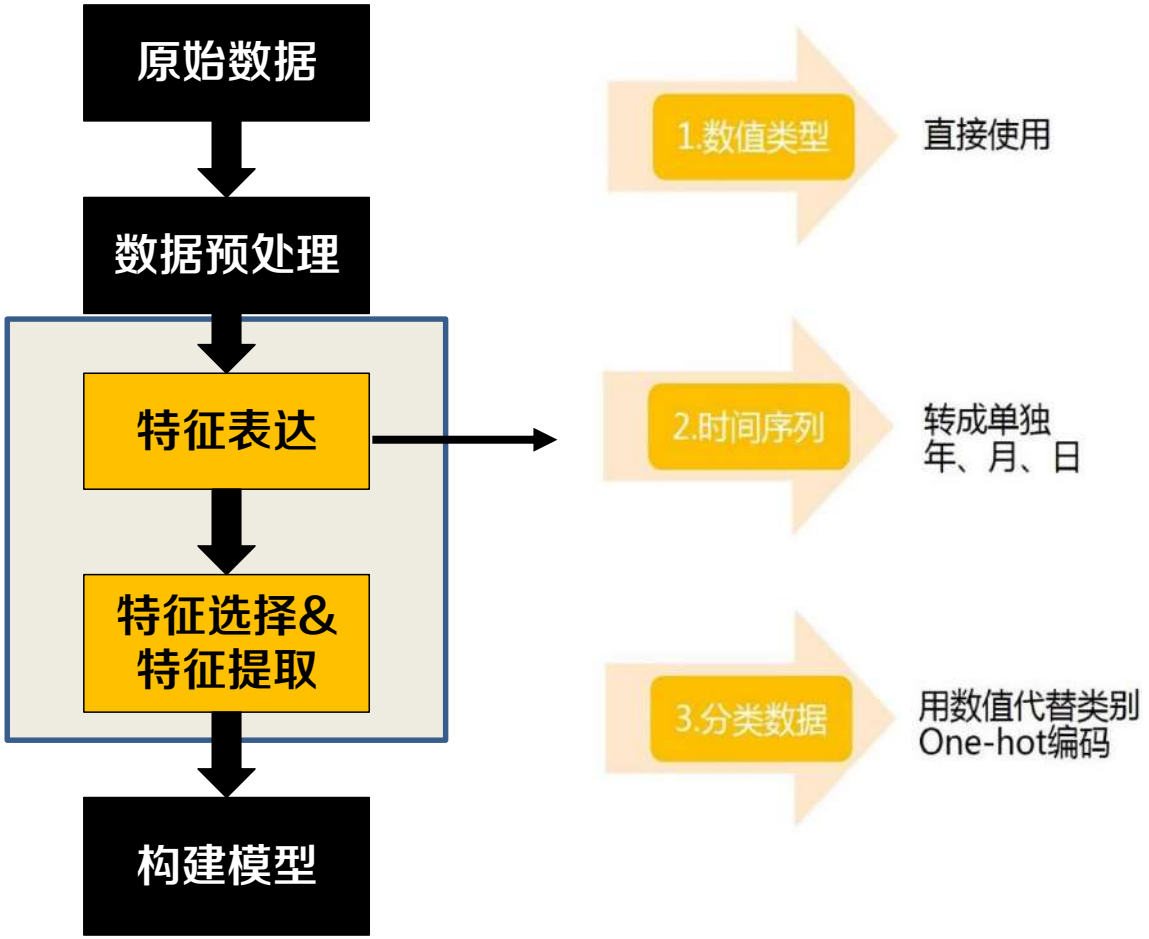
```
#缺失数据比较多，船舱号（Cabin）缺失值填充为U，表示未知（Uknow
full['Cabin'] = full['Cabin'].fillna('U')
```

特征工程

- 什么是特征工程?
- 特征工程就是最大限度地从原始数据中提取特征以供机器学习算法和模型使用
- Feature Engineering is manually designing what the input x’s should be.

特征工程

特征工程处理方法



- One-hot编码
  - 发现并纠正数据文件中可识别的错误的最后一道程序，包括检查数据一致性，处理无效值和缺失值等。

特征工程

数据类型	列名（变量）	含义
标签	Survived	生存情况（1=存活，0=死亡）
数值类型	PassengerId	乘客编号
	Age	年龄
	Fare	船票价格
	SibSp	船上兄妹姐妹数/配偶数（同代直系亲属数）
	Parch	船上父母数/子女数（不同代直系亲属数）
分类数据(有类别)	Sex	性别（男，女）
	Embarked	登船港口 出发地点：S=英国南安普顿Southampton 途径地点1：C=法国 瑟堡市Cherbourg 出发地点2：Q=爱尔兰 昆士敦Queenstown
	Pclass	客舱等级（1=1等舱，2=2等舱，3=3等舱）
分类数据(字符串)	Name	姓名
	Cabin	客舱号
	Ticket	船票编号

分类数据特征表达：性别

```
'''
将性别的值映射为数值
男（male）对应数值1，女（female）对应数值0
'''
sex_mapDict={'male':1,          ①
             'female':0}
#map函数：对Series每个数据应用自定义的函数计算
sex=full['Sex'].map(sex_mapDict)   ②
sex.head()
```

```
0    1
1    0
2    0
3    0
4    1
Name: Sex, dtype: int64
```

特征工程

登船港口


```
#查看该类数据内容
full['Embarked'].head()
```

```
0    S
1    C
2    S
3    S
4    S
```

❶ 原数据

```
#存放提取后的特征
embarkedDf = pd.DataFrame()
#使用get_dummies进行one-hot编码，列名前缀是Embarked
embarkedDf = pd.get_dummies( full['Embarked'] , prefix='Embarked' )
embarkedDf.head()
```

❷ ❸

	Embarked_C	Embarked_Q	Embarked_S
0	0	0	1
1	1	0	0
2	0	0	1
3	0	0	1
4	0	0	1

❹ One-hot 编码后

客舱等级

头等舱（Pclass=1） 二等舱（Pclass=2） 三等舱（Pclass=3）

```
#存放提取后的特征
pclassDF = pd.DataFrame()

#使用get_dummies进行one-hot编码，列名前缀是Pclass
pclassDF = pd.get_dummies( full['Pclass'] , prefix='Pclass' )
pclassDF.head()
```

	Pclass_1	Pclass_2	Pclass_3
0	0	0	1
1	1	0	0
2	0	0	1
3	1	0	0
4	0	0	1

特征工程

数据类型	列名（变量）	含义
分类数据(有类别)	Sex	性别（男，女）
	Embarked	登船港口 出发地点：S=英国南安普顿 Southampton 途径地点1：C=法国 瑟堡市 Cherbourg 出发地点2：Q=爱尔兰 昆士敦Queenstown
	Pclass	客舱等级（1=1等舱，2=2等舱，3=3等舱）
分类数据(字符串)	Name	姓名
	Cabin	客舱号
	Ticket	船票编号

姓名

```
full['Name'].head()
0                              Braund, Mr. Owen Harris
1    Cumings, Mrs. John Bradley (Florence Briggs Th...
2                               Heikkinen, Miss. Laina
3         Futrelle, Mrs. Jacques Heath (Lily May Peel)
4                             Allen, Mr. William Henry
Name: Name, dtype: object
```

name1= 'Braund, Mr. Owen Harris'

字符串格式： 名, 头衔. 姓

```
#Mr. Owen Harris
str1=name1.split( ',' )[1]
#Mr.
str2=str1.split( '.' )[0]
#strip() 方法用于移除字符串头尾指定的字符（默认为空格）
str3=str2.strip()
```

```
'''
定义函数：从姓名中获取头衔
'''
def getTitle(name):
    str1=name.split( ',' )[1] #Mr. Owen Harris
    str2=str1.split( '.' )[0]#Mr
    #strip() 方法用于移除字符串头尾指定的字符（默认为空格）
    str3=str2.strip()
    return str3
```

```
#存放提取后的特征
titleDf = pd.DataFrame()
#map函数：对Series每个数据应用自定义的函数计算
titleDf['Title'] = full['Name'].map(getTitle)
titleDf.head()
```

	Title
0	Mr
1	Mrs
2	Miss
3	Mrs
4	Mr

类别	含义
Officer	政府官员
Royalty	王室（皇室）
Mr	已婚男士
Mrs	已婚妇女
Miss	年轻未婚女子
Master	有技能的人/教师

```
#姓名中头衔字符串与定义头衔类别的映射关系
title_mapDict = {
                    "Capt":       "Officer",
                    "Col":        "Officer",
                    "Major":      "Officer",
                    "Jonkheer":   "Royalty",
                    "Don":        "Royalty",
                    "Sir" :       "Royalty",
                    "Dr":         "Officer",
                    "Rev":        "Officer",
                    "the Countess":"Royalty",
                    "Dona":       "Royalty",
                    "Mme":        "Mrs",
                    "Mlle":       "Miss",
                    "Ms":         "Mrs",
                    "Mr" :        "Mr",
                    "Mrs" :       "Mrs",
                    "Miss" :      "Miss",
                    "Master" :    "Master",
                    "Lady" :      "Royalty"
                    }

#map函数：对Series每个数据应用自定义的函数计算
titleDf['Title'] = titleDf['Title'].map(title_mapDict)

#使用get_dummies进行one-hot编码
titleDf = pd.get_dummies(titleDf['Title'])
titleDf.head()
```

	Master	Miss	Mr	Mrs	Officer	Royalty
0	0	0	1	0	0	0
1	0	0	0	1	0	0
2	0	1	0	0	0	0

特征工程

客舱号

```
#查看客舱号的内容
full['Cabin'].head()
```

```
0       U
1     C85
2       U
3    C123
4       U
Name: Cabin, dtype: object
```

匿名函数语法：
lambda 参数1，参数2：函数体

```
# 定义匿名函数：对两个数相加
sum = lambda a,b: a + b

# 调用sum函数
print ("相加后的值为 : ", sum(10,20))
```

```
#存放客舱号信息
cabinDf = pd.DataFrame()

'''
客场号的类别值是首字母，例如：
C85 类别映射为首字母C
'''
full[ 'Cabin' ] = full[ 'Cabin' ].map( lambda c : c[0] )

#使用get_dummies进行one-hot编码，列名前缀是Cabin
cabinDf = pd.get_dummies( full['Cabin'] , prefix = 'Cabin' )

cabinDf.head()
```

	Cabin_A	Cabin_B	Cabin_C	Cabin_D	Cabin_E	Cabin_F	Cabin_G	Cabin_T	Cabin_U
0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	1
3	0	0	1	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	1

家庭类别

列名（变量）	含义	变量值
SibSp	船上兄妹姐妹数/配偶数（同代直系亲属数）	2
Parch	船上父母数/子女数（不同代直系亲属数）	1

```
#存放家庭信息
familyDf = pd.DataFrame()

'''
家庭人数=同代直系亲属数（Parch）+不同代直系亲属数（SibSp）+乘客自己
（因为乘客自己也是家庭成员的一个，所以这里加1）
'''
familyDf[ 'FamilySize' ] = full[ 'Parch' ] + full[ 'SibSp' ] + 1
```

```
'''
家庭类别：
小家庭Family_Single：家庭人数=1
中等家庭Family_Small：2<=家庭人数<=4
大家庭Family_Large：家庭人数>=5
'''
#if 条件为真的时候返回if前面内容，否则返回0
familyDf[ 'Family_Single' ] = familyDf[ 'FamilySize' ].map( lambda s : 1 if s == 1 else 0 )
familyDf[ 'Family_Small' ]  = familyDf[ 'FamilySize' ].map( lambda s : 1 if 2 <= s <= 4 else 0 )
familyDf[ 'Family_Large' ]  = familyDf[ 'FamilySize' ].map( lambda s : 1 if 5 <= s else 0 )

familyDf.head()
```

	FamilySize	Family_Single	Family_Small	Family_Large
0	2	0	1	0
1	2	0	1	0
2	1	1	0	0
3	2	0	1	0
4	1	1	0	0

特征工程

- **特征选择 feature Selection**
  - 单纯地从提取到的所有特征中选择部分特征作为训练集特征
  - 方法：Principal Component Analysis(主成分分析)、Singular Value Decomposition(奇异值分解) 、Sammon's Mapping(Sammon映射)
- **特征提取 feature extraction**
  - 从一个维度空间映射到另一个维度空间，本质上是降维
  - 方法：Chi-squared test(卡方检验)、information gain(信息增益)、 correlation coefficient scores(相关系数)
- **相同点和不同点**
  - 效果相同：减少特征数据集中的属性(或者称为特征)的数目
  - 特征提取：通过属性间的关系，如组合不同的属性得新的属性，改变了原来的特征空间；
  - 特征选择：是一种包含的关系，没有更改原始的特征空间。

特征工程

- **特征提取结果** (correlation coefficient scores)

```
#相关性矩阵
corrDf = full.corr()
corrDf
```

❶

	Age	Fare	Parch	Sex
Age	1.000000	0.171521	-0.130872	0.057397
Fare	0.171521	1.000000	0.221522	-0.185484
Parch	-0.130872	0.221522	1.000000	-0.213125
Sex	0.057397	-0.185484	-0.213125	1.000000
SibSp	-0.190747	0.160224	0.373587	-0.109609
Survived	-0.070323	0.257307	0.081629	-0.543351

❷

```
'''
查看各个特征与生成情况（Survived）的相关系数，
ascending=False表示按降序排列
'''
corrDf['Survived'].sort_values(ascending =Fals
```

```
Survived         1.000000
Mrs              0.344935
Miss             0.332795
Pclass_1         0.285904
Family_Small     0.279855
Fare             0.257307
```

正线性相关

	Survived	1.000000
	Mrs	0.344935
头衔	Miss	0.332795
客舱等级	Pclass_1	0.285904
家庭大小	Family_Small	0.279855
船票价格	Fare	0.257307
船舱号	Cabin_B	0.175095
	Embarked_C	0.168240
登船港口	Embarked_C	0.168240
	Cabin_D	0.150716
	Cabin_E	0.145321
	Cabin_C	0.114652
	Pclass_2	0.093349
	Master	0.085221
	Parch	0.081629
	Cabin_F	0.057935
	Royalty	0.033391
	Cabin_A	0.022287
	FamilySize	0.016639
	Cabin_G	0.016040
	Embarked_Q	0.003650
	Embarked_Q	0.003650

负线性相关

	Cabin_T	-0.026456
	Officer	-0.031316
	SibSp	-0.035322
	Age	-0.070323
	Family_Large	-0.125147
登船港口	Embarked_S	-0.149683
家庭大小	Family_Single	-0.203367
船舱号	Cabin_U	-0.316912
客舱等级	Pclass_3	-0.322308
性别	Sex	-0.543351
头衔	Mr	-0.549199

特征选择

- **特征选择结果**

```
#特征选择
full_X = pd.concat( [titleDf,#头衔
                     pclassDf,#客舱等级
                     familyDf,#家庭大小
                     full['Fare'],#船票价格
                     cabinDf,#船舱号
                     embarkedDf,#登船港口
                     full['Sex']#性别
                    ] , axis=1 )
full_X.head()
```

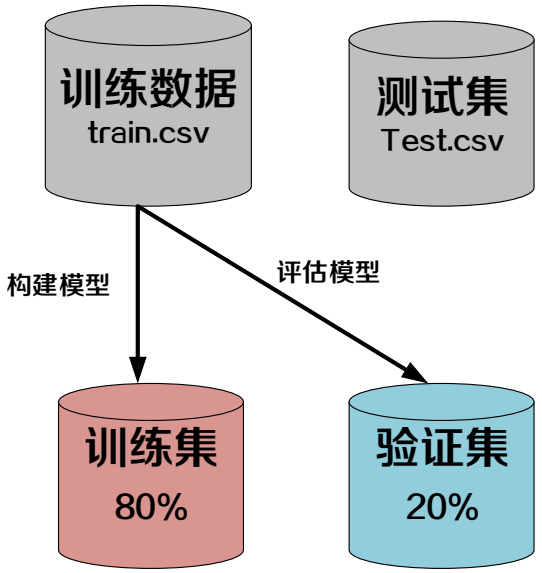
	Master	Miss	Mr	Mrs	Officer	Royalty	Pclass_
0	0	0	1	0	0	0	
1	0	0	0	1	0	0	
2	0	1	0	0	0	0	
3	0	0	0	1	0	0	
4	0	0	1	0	0	0	

5 rows × 27 columns

构建模型

1. 提出问题
2. 理解数据
   1.采集数据
   2.导入数据
   3.查看数据集信息
   describe描述统计信息
   info发现缺失数据
3. 数据清洗
   1.数据预处理
   缺失值填充fillna
   删除缺失值dropna
   2.特征工程
   特征提取：
   One-hot编码
   (get_dummies)
   map函数（series）
   特征选择：相关系数法
4. 构建模型
5. 模型评估
6. 方案实施

构建模型




```
#原始数据集有891行
sourceRow=891
#原始数据集：特征
source_X = full_X.loc[0:sourceRow-1,:]
#原始数据集：标签
source_y = full.loc[0:sourceRow-1,'Survived']

#预测数据集：特征
pred_X = full_X.loc[sourceRow:,:]
```

构建模型

训练数据和测试数据

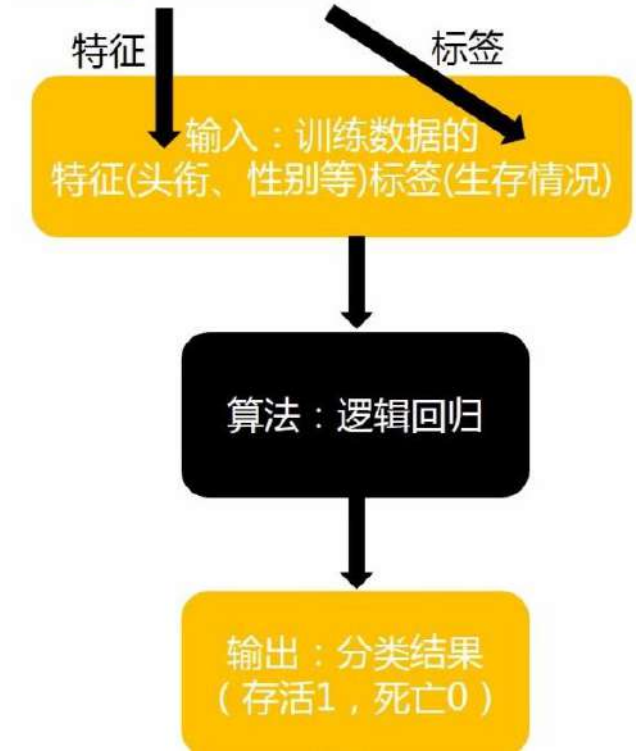
```
from sklearn.cross_validation import train_test_split

#建立模型用的训练数据集和测试数据集
train_X, test_X, train_y, test_y = train_test_split(source_X , 原始数据特征
                                                    source_y,  原始数据标签
                                                    train_size=.8) 训练数据占比
#输出数据集大小
print ('原始数据集特征：',source_X.shape,
       '训练数据集特征：',train_X.shape ,
      '测试数据集特征：',test_X.shape)

print ('原始数据集标签：',source_y.shape,
       '训练数据集标签：',train_y.shape ,
      '测试数据集标签：',test_y.shape)
```

训练数据标签 3　测试数据标签　训练数据特征　测试数据特征　1　2

```
原始数据集特征： (891, 27) 训练数据集特征： (712, 27) 测试数据集特征： (179, 27
原始数据集标签： (891,) 训练数据集标签： (712,) 测试数据集标签： (179,)
```

训练模型

```
#第1步：导入算法
from sklearn.linear_model import LogisticRegression
#第2步：创建模型：逻辑回归（logisic regression）
model = LogisticRegression()
#第3步：训练模型
model.fit( train_X , train_y )
```



模型评估

```
# 分类问题，score得到的是模型的正确率
model.score(test_X , test_y )
```

0.8435754189944134

多个特征	生存情况
x	0
x	1
x	1
x	0

输入：测试数据

模型：逻辑回归算法

输出：预测结果

多个特征	预测结果
x	1
x	0
x	0
x	0

正确率 = 正确分类个数 / 数据总数 ≈ 0%

1. 提出问题
2. 理解数据
   - 1.采集数据
   - 2.导入数据
   - 3.查看数据集信息
   - describe描述统计信息
   - info发现缺失数据
3. 数据清洗
   - 1.数据预处理
   - 缺失值填充fillna
   - 删除缺失值dropna
   - 2.特征工程
   - 特征提取：
   - One-hot编码 (get_dummies)
   - map函数（series）
   - 特征选择：相关系数法
4. 构建模型
   - 1.训练数据和测试数据
   - train_test_split
   - 2.机器学习算法: Sklearn
5. 模型评估
   - Score
6. 方案实施

方案实施

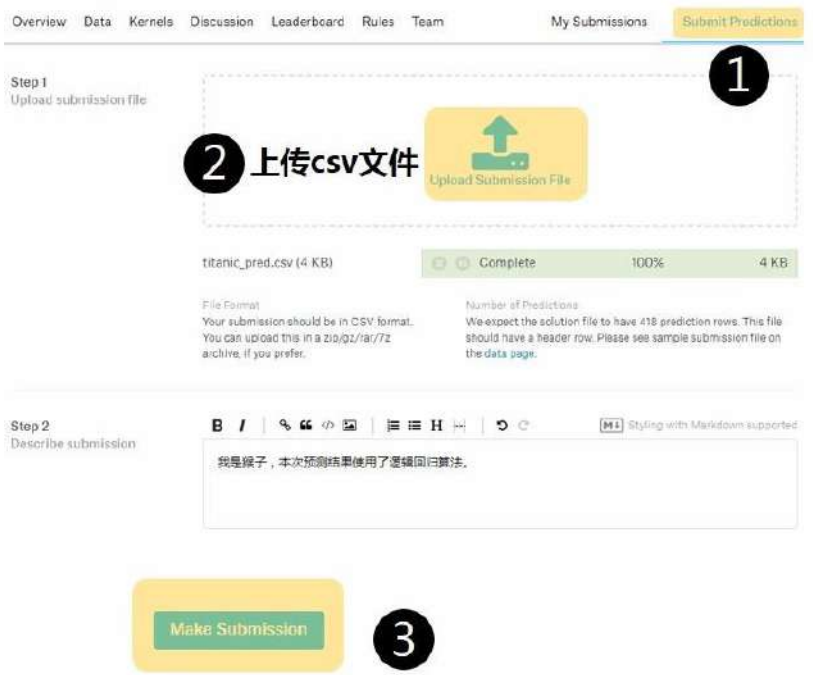
6 方案实施

```
#使用机器学习模型，对预测数据集中的生存情况进行预测
pred_Y = model.predict(pred_X)
'''
生成的预测值是浮点数（0.0,1.0）
但是Kaggle要求提交的结果是整型（0,1）
所以要对数据类型进行转换
'''
pred_Y=pred_Y.astype(int)
#乘客id
passenger_id = full.loc[sourceRow:,'PassengerId']
#数据框：乘客id，预测生存情况的值
predDf = pd.DataFrame(
    {'PassengerId': passenger_id ,
     'Survived': pred_Y } )
predDf.shape
predDf.head()
#保存结果
predDf.to_csv( 'titanic_pred.csv' , index = False )
```

1 2 3

titanic_pred.csv

	PassengerId	Survived
1		
2	892	0
3	893	1
4	894	0
5	895	0



方案实施

1. **提出问题**

2. **理解数据**
   1.采集数据
   2.导入数据
   3.查看数据集信息
   describe描述统计信息
   info发现缺失数据

3. **数据清洗**
   1.数据预处理
   缺失值填充fillna
   删除缺失值dropna
   2.特征工程
   特征提取：
   One-hot编码
   (get_dummies)
   map函数（series）
   特征选择：相关系数法

4. **构建模型**
   1.训练数据和测试数据
   train_test_split
   2.机器学习算法: Sklearn

5. **模型评估**
   Score

6. **方案实施**
   1.提交结果到kaggle
   2.报告撰写

参考文献

参考文献

[1] Megan Risdal.Exploring Survival on the Titanic [R/OL]. https://www.kaggle.com/mrisdal/exploring-survival-on-the-titanic/comments. 6 March 2016.
[2] Manav Sehgal.Titanic Data Science Solutions[R/OL]. https://www.kaggle.com/startupsci/titanic-data-science-solutions/comments. 29 Jan 2017.

道德经

知人者智，自知者明。胜人者有力，自胜者强。知足者富。强行者有志。不失其所者久。死而不亡者，寿。

谢谢！