泰坦尼克号生存预测特征工程

一、数据预处理

通过数据质量分析，检查原始数据中是否存在脏数据，脏数据包括：缺失值，异常值，不一致的值，重复数据及含有特殊符号（如#、￥、*）的值。

1、缺失值分析

将训练集和测试集合并，统一进行数据预处理

```
In [1]:
import numpy as np
import pandas as pd
train=pd.read_csv('./desktop/all/train.csv')#读入训练数据集
test=pd.read_csv('./desktop/all/test.csv')#读入测试数据集
print('训练数据集规模',train.shape)
print('测试数据集规模',test.shape)
```

```
训练数据集规模 (891, 12)
测试数据集规模 (418, 11)
```

```
In [2]:
df=train.append(test,sort=True) #合并数据集，方便进行数据预处理
```

```
In [3]:
print('合并后的数据集规模',df.shape)
```

```
合并后的数据集规模 (1309, 12)
```

训练数据集的规模是 891*12（即 891 条数据，12 个属性）；测试数据集的规模 418*11 属性，测试集少一个属性 Survived，这是需要通过模型预测的；合并后的数据集规模是 1309*12。。

查看前 10 条数据了解一下数据集的格式：

```
In [15]:
df.head(10)
```

Out[15]:

	Age	Cabin	Embarked	Fare	Name	Parch	PassengerId	Pclass	Sex	SibSp	Survived	Ticket
0	22.000	NaN	S	7.250	Braund, Mr. Owen Harris	0	1	3	male	1	0.000	A/5 21171
1	38.000	C85	C	71.283	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	2	1	female	1	1.000	PC 17599
2	26.000	NaN	S	7.925	Heikkinen, Miss. Laina	0	3	3	female	0	1.000	STON/O2. 3101282
3	35.000	C123	S	53.100	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	4	1	female	1	1.000	113803
4	35.000	NaN	S	8.050	Allen, Mr. William Henry	0	5	3	male	0	0.000	373450
5	nan	NaN	Q	8.458	Moran, Mr. James	0	6	3	male	0	0.000	330877
6	54.000	E46	S	51.862	McCarthy, Mr. Timothy J	0	7	1	male	0	0.000	17463
7	2.000	NaN	S	21.075	Palsson, Master. Gosta Leonard	1	8	3	male	3	0.000	349909
8	27.000	NaN	S	11.133	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	2	9	3	female	0	1.000	347742
9	14.000	NaN	C	30.071	Nasser, Mrs. Nicholas (Adele Achem)	0	10	2	female	1	1.000	237736

查看基本的统计信息：

```
In [4]:
pd.options.display.float_format='{:,.3f}'.format #数据显示模式
df.describe()#查看数据的基本统计信息
```

Out[4]:

	Age	Fare	Parch	PassengerId	Pclass	SibSp	Survived
count	1,046.000	1,308.000	1,309.000	1,309.000	1,309.000	1,309.000	891.000
mean	29.881	33.295	0.385	655.000	2.295	0.499	0.384
std	14.413	51.759	0.866	378.020	0.838	1.042	0.487
min	0.170	0.000	0.000	1.000	1.000	0.000	0.000
25%	21.000	7.896	0.000	328.000	2.000	0.000	0.000
50%	28.000	14.454	0.000	655.000	3.000	0.000	0.000
75%	39.000	31.275	0.000	982.000	3.000	1.000	1.000
max	80.000	512.329	9.000	1,309.000	3.000	8.000	1.000

用 df.info()查看每一列的数据总数和数据类型，对于每个属性，合理的数据总数应该是 1309。

```
In [5]: df.info()#查看每一列的数据以及数据类型，进行缺失值分析

<class 'pandas.core.frame.DataFrame'>
Int64Index: 1309 entries, 0 to 417
Data columns (total 12 columns):
Age            1046 non-null float64
Cabin          295 non-null object
Embarked       1307 non-null object
Fare           1308 non-null float64
Name           1309 non-null object
Parch          1309 non-null int64
PassengerId    1309 non-null int64
Pclass         1309 non-null int64
Sex            1309 non-null object
SibSp          1309 non-null int64
Survived       891 non-null float64
Ticket         1309 non-null object
dtypes: float64(3), int64(4), object(5)
memory usage: 132.9+ KB
```

进行缺失值的分析：

属性	含义	数据总数	缺失值	缺失率
Age	年龄	1046	263	20.09%
Cabin	船舱号	295	1014	77.46%
Embarked	登船港口	1307	2	0.15%
Fare	船票价格	1308	1	0.08%
Name	姓名	1309	0	0
Parch	父母/子女数目	1309	0	0
PassengerID	乘客编号	1309	0	0
Pclass	乘客等级（三个等级，1=一等，2=二等，3=三等）	1309	0	0
Sex	性别(男为 male,女为 female)	1309	0	0
SibSp	兄弟姐妹/配偶数目	1309	0	0
Ticket	船票号码	1309	0	0
Survived	是否存活（存活是 1，死亡是 0）	891		

进行缺失值的处理：有缺失值的 Age、Cabin、Embarked、Fare。

年龄 Age 是数值型，缺失值采用平均值填充。

船舱号 Cablin 缺失较多，缺失值可能是包含某种隐藏的信息，比如本来就没有船舱号码，因此处理方法是直接填充为'U',代表 Unkown。

Embarked 缺失值较少，且是分类数据，可以采用众数填充，众数是'S''。

Fare 是船票价格，数值型数据，只缺失一个数据，可以用平均值填充。

```
In [7]: df['Age']=df['Age'].fillna(df['Age'].mean())#年龄缺失值用均值填充
        df['Cabin']=df['Cabin'].fillna('U')#Cbain缺失值用 'U' 填充，代表Unknown
        df['Fare']=df['Fare'].fillna(df['Fare'].mean())#Fare缺失值用均值填充
        print(Counter(df['Embarked']))#Embarked的类别有三类：S、C、Q，众数 'S'

        Counter({'S': 914, 'C': 270, 'Q': 123, nan: 2})

In [9]: df['Embarked']=df['Embarked'].fillna('S')#Embarked缺失值用众数'S'填充
```

缺失值处理之后的数据信息如下：

```
In [9]: df['Embarked']=df['Embarked'].fillna('S')#Embarked缺失值用众数'S'填充

In [11]: #查看缺失值处理之后的数据
         df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1309 entries, 0 to 417
Data columns (total 12 columns):
Age            1309 non-null float64
Cabin          1309 non-null object
Embarked       1309 non-null object
Fare           1309 non-null float64
Name           1309 non-null object
Parch          1309 non-null int64
PassengerId    1309 non-null int64
Pclass         1309 non-null int64
Sex            1309 non-null object
SibSp          1309 non-null int64
Survived       891 non-null float64
Ticket         1309 non-null object
dtypes: float64(3), int64(4), object(5)
memory usage: 132.9+ KB
```

二、特征工程

特征分析，共 11 个特征：

数值类型：Age、Fare、SibSp、Parch、PassergerID

分类类型：Pclass、Sex、Embarked

字符串类型：Name、Cabin、Ticket

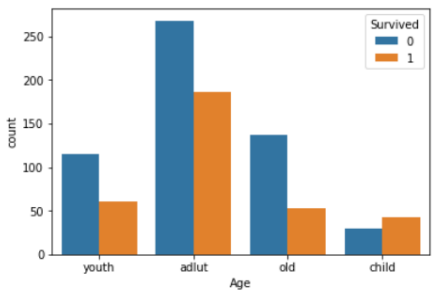
数值类型可以直接使用，分类类型用数值代替类别（one-hot 编码），同时由于数据量较小，一些明显无关的特征可以删除，防止过拟合。

1、 Age

由于 Age 是连续性变量，不好观察特征，按照国际的标准将年龄划分为 4 类，0-14 岁为 child，15-24 岁为 youth，25-64 为 adult，大于 64 岁为 old，可以看到明显老人和小孩存活下来的可能性更大。Age 特征明显影响存活率，要保留该项特征。

```
In [10]: #年龄特征分类
         train['Age']=train['Age'].map(lambda x: 'child' if x<14 else 'youth' if x<24 else 'adlut' if x<64 else 'old' )
         sns.countplot(x='Age',hue="Survived",data=train)
```

```
Out[10]: <matplotlib.axes._subplots.AxesSubplot at 0x23f8eaed1d0>
```



分类，进行 one-hot 编码：

In [11]:
```
AgeDf=pd.DataFrame()
AgeDf['child']=df['Age'].map(lambda x:1 if x<14 else 0)
AgeDf['youth']=df['Age'].map(lambda x:1 if 14<x<24 else 0)
AgeDf['adult']=df['Age'].map(lambda x:1 if 24<x<64 else 0)
AgeDf['old']=df['Age'].map(lambda x:1 if x>64 else 0)
AgeDf.head()
```

Out[11]:

	child	youth	adult	old
0	0	1	0	0
1	0	0	1	0
2	0	0	1	0
3	0	0	1	0
4	0	0	1	0

In [12]:
```
df=pd.concat([df,AgeDf],axis=1)#将one-hot产生的虚拟变量加入到数据集中
df.drop('Age',axis=1,inplace=True)#删除原来的Age属性
df.head(5)
```

Out[12]:

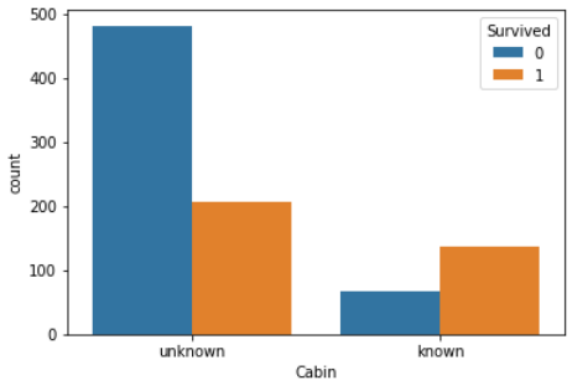
	Cabin	Embarked	Fare	Name	Parch	PassengerId	Pclass	Sex	SibSp	Survived	Ticket	child	youth	adult	old
0	U	S	7.250	Braund, Mr. Owen Harris	0	1	3	male	1	0.000	A/5 21171	0	1	0	0
1	C85	C	71.283	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	2	1	female	1	1.000	PC 17599	0	0	1	0
2	U	S	7.925	Heikkinen, Miss. Laina	0	3	3	female	0	1.000	STON/O2. 3101282	0	0	1	0
3	C123	S	53.100	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	4	1	female	1	1.000	113803	0	0	1	0
4	U	S	8.050	Allen, Mr. William Henry	0	5	3	male	0	0.000	373450	0	0	1	0

2、 Cabin

Cablin 数据有很多缺失值，可以分为两类，数据存在和数据不存在，可以看到这两种类型的存活率有一定的差别，可以认为 Cablin 对存活与否也是有影响的，并且可以将缺失值作为一种新的类型，即前面数据预处理的将缺失值填充为'U'。

In [8]:
```
train['Cabin']=train['Cabin'].map(lambda x:'known' if type(x)==str else 'unknown')
sns.countplot(x='Cabin',hue="Survived",data=train)
```

Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0x27d6c50d1d0>



观察数据可以看到港口位置可以用第一个字符代表，可以分为 A、B、C、D、E、F、G、T、U 九类。使用 one-hot 编码。

In [24]:
```
cabinDf=pd.DataFrame()#创建pandas中的DataFrame对象
df['Cabin']=df['Cabin'].map(lambda x:x[0])
cabinDf=pd.get_dummies(df['Cabin'],prefix='Cabin')#使用get_dummies进行one-hot编码，列名前缀是Cabin
cabinDf.head()
```

Out[24]:

	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

```
In [15]: df=pd.concat([df,cabinDf],axis=1)#将one-hot产生的虚拟变量加入到数据集中
         df.drop('Cabin',axis=1,inplace=True)#删除原来的Cabin属性
         df.head(5)
```

Out[15]:

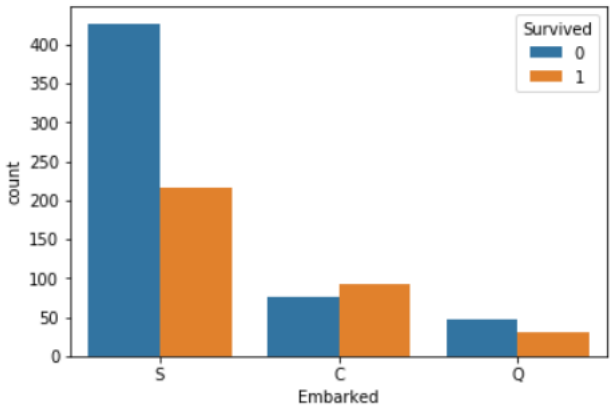
	Age	Embarked	Fare	Name	Parch	PassengerId	Pclass	Sex	SibSp	Survived	Ticket	Cabin_A	Cabin_B	Cabin_C	Cabin_D	Cabin_E	Cab
0	22.000	S	7.250	Braund, Mr. Owen Harris	0	1	3	1	1	0.000	A/5 21171	0	0	0	0	0	
1	38.000	C	71.283	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	2	1	0	1	1.000	PC 17599	0	0	1	0	0	
2	26.000	S	7.925	Heikkinen, Miss. Laina	0	3	3	0	0	1.000	STON/O2. 3101282	0	0	0	0	0	
3	35.000	S	53.100	Futrelle, Mrs. Jacques Heath (Lily May	0	4	1	0	1	1.000	113803	0	0	1	0	0	

3、 Embarked

Embarked 共有三类，分别是 S、C、Q,存活的情况有一定差别，C 港口的生存概率大一些，此特征保留。

```
In [10]: sns.countplot(x='Embarked',hue="Survived",data=train)
```

Out[10]: <matplotlib.axes._subplots.AxesSubplot at 0x2b375b8bef0>



使用 One-hot 编码

```
In [16]: EmbarkedDf=pd.DataFrame()#创建pandas中的DataFrame对象
         df['Embarked']=df['Embarked'].map(lambda x:x[0])
         EmbarkedDf=pd.get_dummies(df['Embarked'],prefix='Embarked')#使用get_dummies进行one-hot编码，列名前缀是Embarked
         EmbarkedDf.head()
```

Out[16]:

	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

In [17]:
```
df=pd.concat([df,EmbarkedDf],axis=1)#将one-hot产生的虚拟变量加入到数据集中
df.drop('Embarked',axis=1,inplace=True)#删除原来的Embarked属性
df.head(5)
```

Out[17]:

	Age	Fare	Name	Parch	PassengerId	Pclass	Sex	SibSp	Survived	Ticket	...	Cabin_C	Cabin_D	Cabin_E	Cabin_F	Cabin_G	Cabin_T	Cab
0	22.000	7.250	Braund, Mr. Owen Harris	0	1	3	1	1	0.000	A/5 21171	...	0	0	0	0	0	0	
1	38.000	71.283	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	2	1	0	1	1.000	PC 17599	...	1	0	0	0	0	0	
2	26.000	7.925	Heikkinen, Miss. Laina	0	3	3	0	0	1.000	STON/O2. 3101282	...	0	0	0	0	0	0	
3	35.000	53.100	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	4	1	0	1	1.000	113803	...	1	0	0	0	0	0	
4	35.000	8.050	Allen, Mr. William Henry	0	5	3	1	0	0.000	373450	...	0	0	0	0	0	0	

5 rows × 22 columns

4、 Fare

Fare 是船票的价格，在小于 2.5 和大于 2.5 时候存活率有明显的差异

One-hot 编码

In [34]:
```
FareDf=pd.DataFrame()
FareDf['Fare_low']=df['Fare'].map(lambda x:1 if x<=2.5 else 0)
FareDf['Fare_high']=df['Fare'].map(lambda x:1 if x>2.5 else 0)
FareDf.head()
```

Out[34]:

	Fare_low	Fare_high
0	0	1
1	0	1
2	0	1
3	0	1
4	0	1

In [35]:
```
df=pd.concat([df,FareDf],axis=1)#将one-hot产生的虚拟变量加入到数据集中
df.drop('Fare',axis=1,inplace=True)#删除原来的属性
df.head(5)
```

Out[35]:

t	child	youth	adult	old	Cabin_A	...	Pclass_2	Pclass_3	Parch_none	Parch_less	Parch_many	SibSp_none	SibSp_less	SibSp_many	Fare_low	Fare_high
1	0	1	0	0	0	...	0	1	1	0	0	0	1	0	0	1
9	0	0	1	0	0	...	0	0	1	0	0	0	1	0	0	1
2	0	0	1	0	0	...	0	1	1	0	0	1	0	0	0	1
3	0	0	1	0	0	...	0	0	1	0	0	0	1	0	0	1
0	0	0	1	0	0	...	0	1	1	0	0	1	0	0	0	1

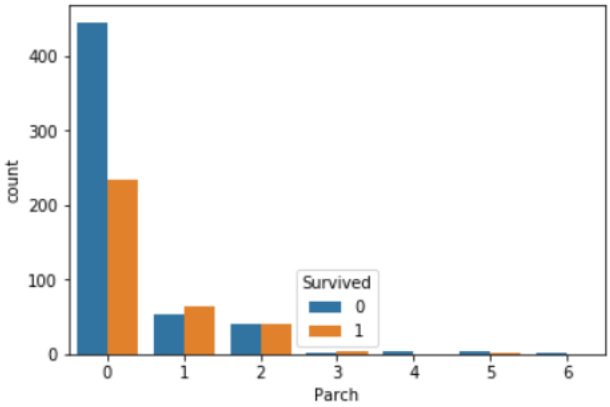
5、 Name

Name 中可能包含一些重要的信息，比如人的称呼、头衔。本模型为了简便，暂时舍弃。后续改进特征工程时再进行处理。

6、 Parch

Parch 表示父母/子女数目，可以看到有父母/子女陪同的，比单独出行的生存率高，此特征也需要保留。

```
In [24]: sns.countplot(x="Parch", hue="Survived", data=train)
Out[24]: <matplotlib.axes._subplots.AxesSubplot at 0x2b37616fa20>
```



分为三个类别：Parch 为 0、Parch 为 1 到 4、Parch 大于 4，进行 one-hot 编码

```
In [25]: ParchDf=pd.DataFrame()
         ParchDf['Parch_none']=df['Parch'].map(lambda x:1 if x==0 else 0)
         ParchDf['Parch_less']=df['Parch'].map(lambda x:1 if 1<=x<=4 else 0)
         ParchDf['Parch_many']=df['Parch'].map(lambda x:1 if x>4 else 0)
         ParchDf.head()
```

Out[25]:

	Parch_none	Parch_less	Parch_many
0	1	0	0
1	1	0	0
2	1	0	0
3	1	0	0
4	1	0	0

```
In [26]: df=pd.concat([df,ParchDf],axis=1)#将one-hot产生的虚拟变量加入到数据集中
         df.drop('Parch',axis=1,inplace=True)#删除原来的Parch属性
         df.head(5)
```

Out[26]:

ived	Ticket	child	youth	adult	...	Cabin_U	Embarked_C	Embarked_Q	Embarked_S	Pclass_1	Pclass_2	Pclass_3	Parch_none	Parch_less	Parch_many
.000	A/5 21171	0	1	0	...	1	0	0	1	0	0	1	1	0	0
.000	PC 17599	0	0	1	...	0	1	0	0	1	0	0	1	0	0
.000	STON/O2. 3101282	0	0	1	...	1	0	0	1	0	0	1	1	0	0
.000	113803	0	0	1	...	0	0	0	1	1	0	0	1	0	0
.000	373450	0	0	1	...	1	0	0	1	0	0	1	1	0	0

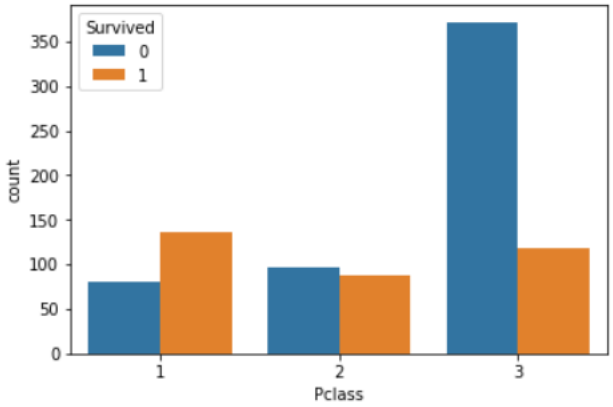
7、 PassengerID

乘客的 ID，ID 是无关信息，删除。

8、 Pclass

可以看到社会等级高的阶层存活概率显然更大，此特征对生存概率有很大影响，需要保留。

```
In [13]: sns.countplot(x='Pclass',hue="Survived",data=train)
Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x21c69bde5f8>
```



Pclass 为 1、2、3 类，使用 one-hot 编码

```
In [18]: PclassDf=pd.DataFrame()#创建pandas中的DataFrame对象
         PclassDf=pd.get_dummies(df['Pclass'],prefix='Pclass')#使用get_dummies进行oone-hot编码，列名前缀是Pclass
         PclassDf.head()
```

Out[18]:

	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

```
In [19]: df=pd.concat([df,PclassDf],axis=1)#将one-hot产生的虚拟变量加入到数据集中
         df.drop('Pclass',axis=1,inplace=True)#删除原来的Pclass属性
         df.head(5)
```

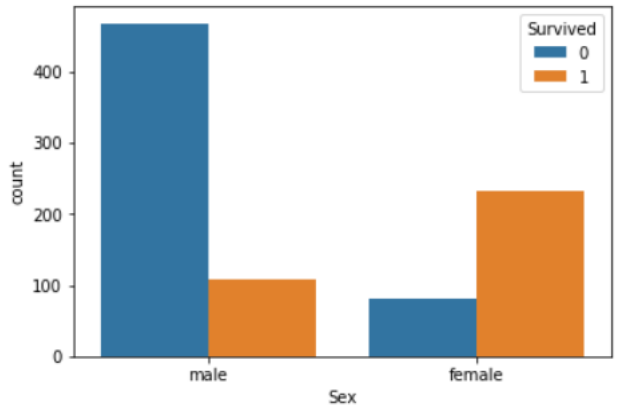
Out[19]:

	Sex	SibSp	Survived	Ticket	Cabin_A	...	Cabin_F	Cabin_G	Cabin_T	Cabin_U	Embarked_C	Embarked_Q	Embarked_S	Pclass_1	Pclass_2	Pclass_3
	1	1	0.000	A/5 21171	0	...	0	0	0	1	0	0	1	0	0	1
	0	1	1.000	PC 17599	0	...	0	0	0	0	1	0	0	1	0	0
	0	0	1.000	STON/O2. 3101282	0	...	0	0	0	1	0	0	1	0	0	1
	0	1	1.000	113803	0	...	0	0	0	0	0	0	1	1	0	0
	1	0	0.000	373450	0	...	0	0	0	1	0	0	1	0	0	1

9、 Sex

女性的存活概率更大，性别特征对生存率有影响，此项特征保留。

```
In [14]: sns.countplot(x='Sex',hue="Survived",data=train)
Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x21c69fafb70>
```



由于 Sex 是分类数据，用数值代替类别，1 代表 male，0 代表 female,进行编码

```
In [12]: sex_mapDict={"male":1,"female":0}
df.Sex=df.Sex.map(sex_mapDict)
df.head(5)
```

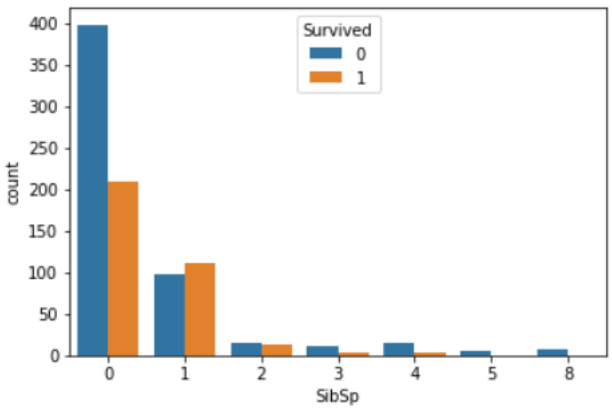
Out[12]:

	Age	Cabin	Embarked	Fare	Name	Parch	PassengerId	Pclass	Sex	SibSp	Survived	Ticket
0	22.000	U	S	7.250	Braund, Mr. Owen Harris	0	1	3	1	1	0.000	A/5 21171
1	38.000	C85	C	71.283	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	2	1	0	1	1.000	PC 17599
2	26.000	U	S	7.925	Heikkinen, Miss. Laina	0	3	3	0	0	1.000	STON/O2. 3101282
3	35.000	C123	S	53.100	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	4	1	0	1	1.000	113803
4	35.000	U	S	8.050	Allen, Mr. William Henry	0	5	3	1	0	0.000	373450

10、 SibSp

和 Parch 类似，分为三类进行 One-hot 编码

```
In [23]: sns.countplot(x="SibSp", hue="Survived", data=train)
Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x2b3761014a8>
```



In [27]:
```
SibSpDf=pd.DataFrame()
SibSpDf['SibSp_none']=df['SibSp'].map(lambda x:1 if x==0 else 0)
SibSpDf['SibSp_less']=df['SibSp'].map(lambda x:1 if 1<=x<=4 else 0)
SibSpDf['SibSp_many']=df['SibSp'].map(lambda x:1 if x>4 else 0)
SibSpDf.head()
```

Out[27]:

	SibSp_none	SibSp_less	SibSp_many
0	0	1	0
1	0	1	0
2	1	0	0
3	0	1	0
4	1	0	0

In [28]:
```
df=pd.concat([df,SibSpDf],axis=1)#将one-hot产生的虚拟变量加入到数据集中
df.drop('SibSp',axis=1,inplace=True)#删除原来的Parch属性
df.head(5)
```

Out[28]:

	Ticket	child	youth	adult	old	...	Embarked_S	Pclass_1	Pclass_2	Pclass_3	Parch_none	Parch_less	Parch_many	SibSp_none	SibSp_less	SibSp_many
	A/5 21171	0	1	0	0	...	1	0	0	1	1	0	0	0	1	0
	PC 17599	0	0	1	0	...	0	1	0	0	1	0	0	0	1	0
	STON/O2. 3101282	0	0	1	0	...	1	0	0	1	1	0	0	1	0	0
	113803	0	0	1	0	...	1	1	0	0	1	0	0	0	1	0
	373450	0	0	1	0	...	1	0	0	1	1	0	0	1	0	0

11、 Ticket

船票号码，无关特征，删除。

最终保留特征：

PclassDf,ParchDf,SibSpDf,FareDf, df.Sex, cabinDf, EmbarkedDf

In [34]:
```
df_X=pd.concat([PclassDf,
                ParchDf,
                SibSpDf,
                FareDf,
                df.Sex,
                cabinDf,
                EmbarkedDf],
                axis=1
                )
df_X.head()
```

Out[34]:

	Pclass_1	Pclass_2	Pclass_3	Parch_none	Parch_less	Parch_many	SibSp_none	SibSp_less	SibSp_many	Fare_low	...	Cabin_C	Cabin_D	Cabin_E	Cal
0	0	0	1	1	0	0	0	1	0	0	...	0	0	0	
1	1	0	0	1	0	0	0	1	0	0	...	1	0	0	
2	0	0	1	1	0	0	1	0	0	0	...	0	0	0	
3	1	0	0	1	0	0	0	1	0	0	...	1	0	0	
4	0	0	1	1	0	0	1	0	0	0	...	0	0	0	

三、特征工程改进思路

1、对于连续值划分为几类，本文没有做出准确的划分，比如 Age 的划分，本文是根据经验划分的年龄段，需要改进；同样的还有 Fare 的划分。

2、Name 特征本文为了简便直接删除，事实上 Name 中包含了头衔、职业等，可以对 Name 特征进行分析并加上这个特征。

3、Cabin 特征是否保留可以再进行对比分析。

4、考虑特征之间的关联。