泰坦尼克号问题的分析报告

日期：2021/12/16　　姓名：刘格磊　　学号：2019204063

题目：泰坦尼克号预测

数据集名：Titanic

针对数据集的描述:

(例如数据集的行、列、索引、数据结构等相关信息的描述，包括但不限于介绍如何采集的数据、从何处采集的数据、用什么方法采集的数据等)

数据集来源：https://www.kaggle.com/c/titanic/data，划到下方点击 down all 即可完成下载

数据集分析：

1. train.csv: 此文件包含船上 891 名乘客的详细信息。首行是每个乘客的乘员 ID、是否幸存、船舱号、姓名、性别、年龄、兄弟姐妹数量、父母子女数量、票号、票价、客舱号、登船港口，接下来的 891 行就是乘客对应属性的信息。(其中的乘员 ID、是否幸存、船舱号、年龄、兄弟姐妹数量、父母子女数量、票价为数字类型，姓名、性别、票号、客舱号、登船港口为字符串类型)

	A	B	C	D	E	F	G	H	I	J	K	L
1	PassengerI	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
2	1	0	3	Braund, Mr	male	22	1	0	A/5 21171	7.25		S
3	2	1	1	Cumings, M	female	38	1	0	PC 17599	71.2833	C85	C
4	3	1	3	Heikkinen,	female	26	0	0	STON/O2.	7.925		S
5	4	1	1	Futrelle, M	female	35	1	0	113803	53.1	C123	S
6	5	0	3	Allen, Mr. \	male	35	0	0	373450	8.05		S
7	6	0	3	Moran, Mr	male		0	0	330877	8.4583		Q
8	7	0	1	McCarthy,	male	54	0	0	17463	51.8625	E46	S
9	8	0	3	Palsson, M	male	2	3	1	349909	21.075		S
10	9	1	3	Johnson, M	female	27	0	2	347742	11.1333		S
11	10	1	2	Nasser, Mr	female	14	1	0	237736	30.0708		C

图 1.train.csv 示意图

2. test.csv: 此文件包含了 kaggle 官方给的其他 418 名乘客除是否幸存这一属性外的详细信息(即乘员 ID、船舱号、姓名、性别、年龄、兄弟姐妹数量、父母子女数量、票号、票价、客舱号、登船港口)，要求我们利用 train.csv 进行建模预测该文件中乘客是否死亡。

	A	B	C	D	E	F	G	H	I	J	K
1	PassengerI	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
2	892	3	Kelly, Mr. J	male	34.5	0	0	330911	7.8292		Q
3	893	3	Wilkes, Mr	female	47	1	0	363272	7		S
4	894	2	Myles, Mr.	male	62	0	0	240276	9.6875		Q
5	895	3	Wirz, Mr. A	male	27	0	0	315154	8.6625		S
6	896	3	Hirvonen, I	female	22	1	1	3101298	12.2875		S
7	897	3	Svensson,	male	14	0	0	7538	9.225		S
8	898	3	Connolly, I	female	30	0	0	330972	7.6292		Q
9	899	2	Caldwell, M	male	26	1	1	248738	29		S
10	900	3	Abrahim, M	female	18	0	0	2657	7.2292		C

图 2.test.csv 示意图

3. gender_submission.csv:此文件是 kaggle 官方给的样例提交，要求最后提交的 csv 文件里面只有乘客编号和你对应乘客编号预测的是否幸存属性。

	A	B
1	PassengerI	Survived
2	892	0
3	893	1
4	894	0
5	895	0
6	896	1
7	897	0
8	898	1
9	899	0
10	900	1

图 3. gender_submission.csv 示意图

分析观点：

观点 1:

对数据的属性进行分析：

PassengerId，乘客的 id 号，由于 id 号是随机分配给乘客的，故可以认为此属性对是否生存没影响，**可以忽略此属性。**

Survived，生存的标号，属性值为 1 表示活了了下来；属性值为 0，则表示死亡。

Pclass，船舱等级，一般船舱等级高安全性也比较高，这个属性会对生产率有影响，**此属性保留。**

Name，名字，很明显此属性对是否生存没影响，**可以忽略此属性。**

Sex，性别，这个因为全球都说 lady first，女士优先，**此属性保留。**

Age，年龄，因为不同年龄体质不同，**此属性保留。**

SibSp，兄弟姐妹数量，有些人和兄弟姐妹一起上船的。这个会有影响，因为有可能因为救他们而导致自己没有上救生船。**此属性保留。**

Parch，父母儿女数量。有些人会带着父母小孩上船的。这个也可能因为要救父母小孩耽误上救生船。**此属性保留。**

Ticket，票的编号。此属性是随机生成的，对是否生存没影响，**可以忽略此属性。**

Fare，票价。这个和 Pclass 有相同的道理，有钱人和权贵比较有势力和影响力。**此属性保留。**

Cabin，舱号。住的舱号与 Ticket 同理，对是否生存没有影响，**可以忽略此属性。**

Embarked，上船港口。不同的上船地方，可能显示人的地位、生活环境、衣着和生活习惯不同。对是否生存有影响，**此属性保留。**

综上所述，我们在处理数据集时，将 PassengerId、Name、Ticket、Cabin 对是否生存无关的属性去除，只保留 Pclass、Sex、Age、SibSp、Parch、Fare、Embarked 这些有关属性。

观点 2:

观察进行第一步处理后的数据集，明显可以看出存在数据丢失的问题，我们对这些缺失的数据进行填充。

Pclass：船舱等级，此属性无丢失，**不做补充处理。**

Sex：性别，此属性无丢失，**不做补充处理。**

Age：年龄，此属性有丢失，**将空缺项填充为年龄的中位数。**

SibSp：兄弟姐妹数量，此属性无丢失，**不做补充处理。**

Parch：父母儿女数量，此属性无丢失，**不做补充处理。**

Fare：票价，此属性有丢失，**将空缺项填充为票价的平均数。**

Embarked：上船港口，很明显上船港口只有三个不同的港口号，**将空缺项填充为出现次数最多的港口号。**

观点 3:

观察进行第二步处理后的数据集，此时我们发现 Sex、Embarked 这俩个属性的值还是字符串类型，还不是数字类型，这对我们进行数据分析处理有障碍，故我们对 Sex、Embarked 这俩个属性进行离散化。

Sex：性别，只有’male’和’female’两个字符串，我们将其分别离散成整数 0 和 1。

Embarked：上船港口，很明显上船港口只有三个不同的港口号：’S’、’C’、’Q’，我们将其分别离散成 0、1、2.

此时，数据的所有处理已经完成，接下来要进行数据分析。

观点 4：

可以用随机森林算法实现构建模型进行预测。

为了获得更好的预测模型，我们先设好随机种子为 56(无特殊意义)，再对不同分簇的模型进行跑分预测，选取其中准确度最高的模型进行最后的建模预测，导出预测数据

out.csv，提交到 kaggle 即可。

观点 5：

此时我们已经完成了对此数据集的建模预测，接下来我们使用相关性分析函数探索是否存活这一属性与其他属性的相关性：

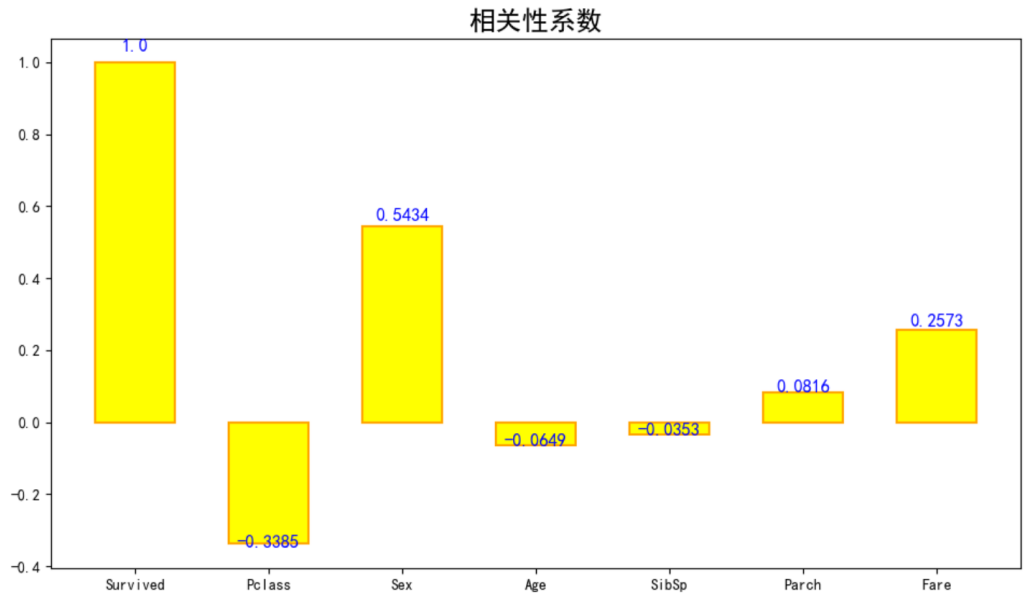

图 4. 是否存活相关性系数表

如图所示，我们可以看出性别、船舱号和票价与是否存活这一属性有很大的相关性，船舱号小的、票价高的、性别为女性的人存活下来的概率更大。

观点 6：

除了对是否存活这一数据进行相关性分析，我们试着对其他的属性也进行分析。分析图如下：

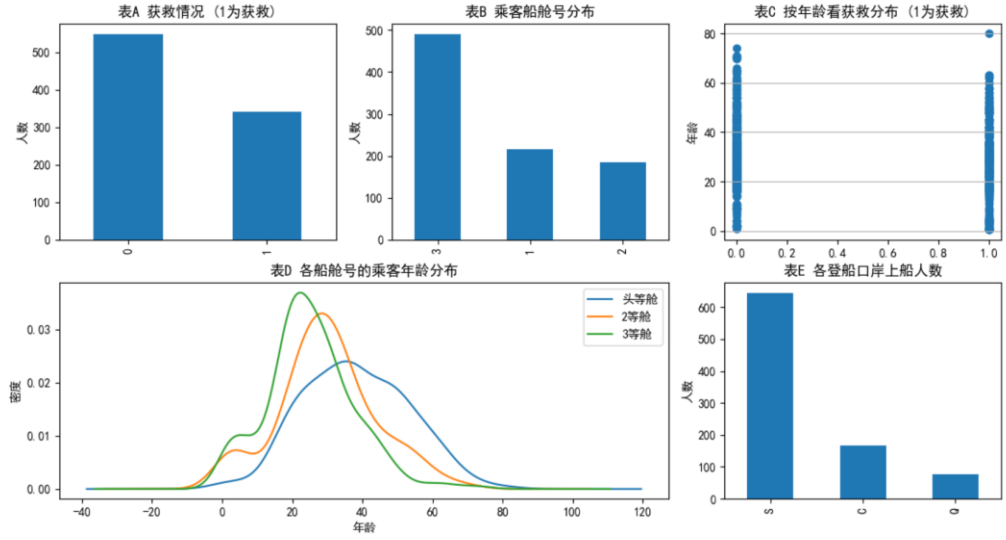

图 5. 其他属性分析图

由图 5 可得：成功存活的人为少数；船舱号为 3 的船人数最多；随船舱号增大年龄密度变小，3 等仓年轻人密度最大；在 S 口上岸的人最多。

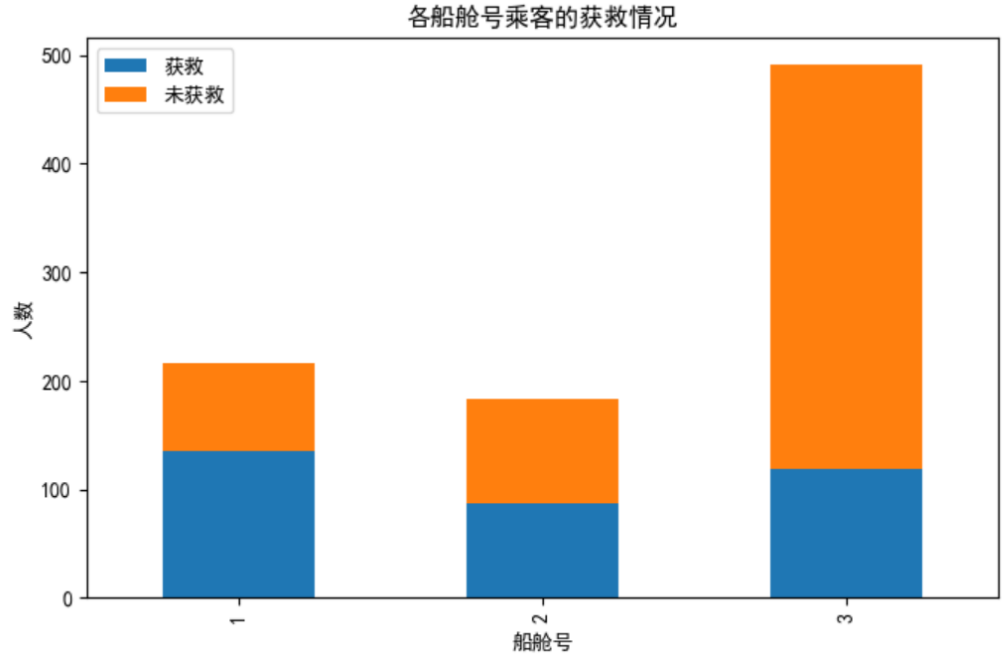

图 6. 各船舱号乘客的获救情况图

由图 6 得知，船舱号 3 的船未获救人数最多且占比最大，而头等舱的获救比例最高。

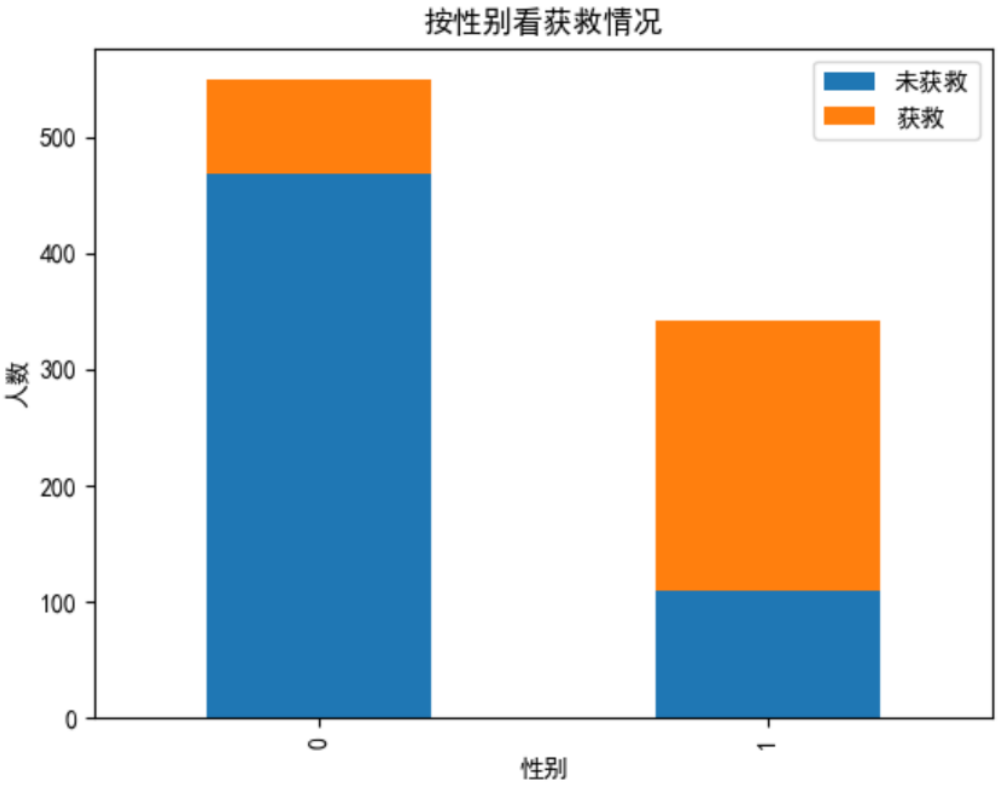

图 7. 按性别看获救情况图

由图 7 可得，女性的获救概率要比男性大的多，女性的获救:未获救大约为 3:1，男性的获救:未获救大约为 1:6.

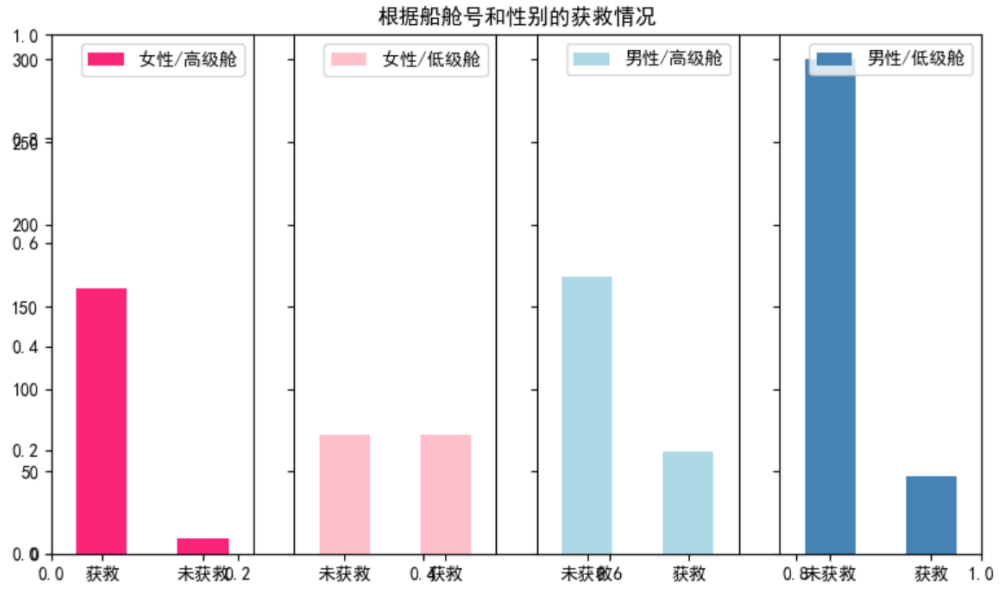

图 8. 根据船舱号和性别的获救情况图

由图 8 可知，在高级舱的女性获救概率极高，而在低级舱的女性获救比例为 0.5；无论是高级舱还是低级舱的男性未获救的概率都远大于获救概率，低级舱内的男性未获救概率更是极高。

分析采用的方法：

随机森林算法：随机森林就是通过集成学习的思想将多棵树集成的一种算法，它的基本单元是决策树，而它的本质属于机器学习的一大分支——集成学习（Ensemble Learning）方法。

通过 train.csv 文件处理后分离出相关属性值集合 X,预测属性值集合 Y，设置随机种子为 56(无特殊意义)，再对不同分簇的模型进行跑分预测，选取其中准确度最高的模型进行最后的建模预测，选取平均精度最高的分簇数，再次使用此模型进行预测。

相关性分析：本题使用 pandas 自带的 corr()函数进行相关系数计算，计算后使用 PIL 库中的 plt 画图函数绘制图形。

代码:

```
#在此处粘贴你的源代码:
#注意格式、字体、字号
# 随机森林算法
import pandas as pd
import matplotlib.pyplot as plt
#导入决策树函数库
from sklearn.tree import DecisionTreeClassifier
#导入 sklearn 中自带的数据集中的 wine，txt 数据集
from sklearn.datasets import load_wine
#导入随机森林函数库
from sklearn.ensemble import RandomForestClassifier
#导入划分训练集和测试集的函数库，交叉验证函数库
from sklearn.model_selection import train_test_split,cross_val_score
#导入精度评分、混淆矩阵函数库
from sklearn.metrics import accuracy_score,confusion_matrix
```

```
# 导入原数据文件
taitan = pd.read_csv(r'D:\作业\数据分析\train.csv')

# 查看一下
df = pd.read_csv(r'D:\作业\数据分析\train.csv', names=['乘客 ID','是否幸存','仓位等级','姓名','性别','年龄','兄弟姐妹个数','父母子女个数','船票信息','票价','客舱','登船港口'],index_col='乘客 ID',header=0)
print(df.head(5))

# 数据预处理
# 将年龄空的设为年龄中位数
taitan["Age"] = taitan["Age"].fillna(taitan["Age"].median())

# 将空缺的性别设为男
taitan["Sex"] = taitan["Sex"].fillna("male")
# 将男女字符串离散成 0 和 1
taitan.loc[taitan["Sex"] == "male", "Sex"] = 0
taitan.loc[taitan["Sex"] == "female", "Sex"] = 1

# 将票价空缺的填为平均值
taitan['Fare'] = taitan['Fare'].fillna(taitan['Fare'].mean())

# 将空缺的登岸港口设为 数量最多的港口
taitan['Embarked'] =
taitan['Embarked'].fillna( taitan['Embarked'].value_counts().index[0])
# 将登岸港口离散化为 0、1、2
taitan.loc[taitan["Embarked"] == "S", "Embarked"] = 0
taitan.loc[taitan["Embarked"] == "C", "Embarked"] = 1
taitan.loc[taitan["Embarked"] == "Q", "Embarked"] = 2
print(taitan.describe())

k_range = range(1, 31)
cv_scores = []
X = taitan.iloc[:,:]

X = X.drop(["PassengerId","Survived","Name","Ticket","Cabin"], axis = 1)
# 输出检查一下
print(X)

Y = taitan.iloc[:, 1]

print(Y)

# 先根据 accuracy 算一下分 确定分簇多少
for n in k_range:
    rfc = RandomForestClassifier(n_estimators = n, random_state = 56, n_jobs = -1)
```

```
    scores = cross_val_score(rfc, X, Y, cv = 100, scoring = 'accuracy', n_jobs = -1)
    cv_scores.append(scores.mean())
plt.plot(k_range, cv_scores)
plt.xlabel('K')
plt.ylabel('Accuracy')
plt.show()
# print("\n")

# 图很明显 簇为 12 时 accuracy 分最高
# 给测试集跑一下分
taitan_list = ['accuracy', 'f1_micro', 'f1_macro', 'f1_weighted']
for s in taitan_list:
    rfc = RandomForestClassifier(n_estimators = 12, random_state = 6, n_jobs = -1)
    scores = cross_val_score(rfc, X, Y, cv = 10, scoring = s , n_jobs = -1)
    print('element = {},  score = {}'.format(s, scores.mean()))

# 导入测验集数据
data_test = pd.read_csv(r'D:\作业\数据分析\test.csv')

# 存储测验集乘客 ID
ID = data_test['PassengerId']

df = data_test.drop(["PassengerId","Name","Ticket","Cabin"], axis = 1)
# 用平均值填充空值
df['Age'] = df['Age'].fillna(df['Age'].median())
df['Fare'] = df['Fare'].fillna(df['Fare'].mean())
# 用数量最多项填充


# data_test = df[len(taitan):]
# 将空缺的性别设为男
df["Sex"] = df["Sex"].fillna("male")
# 将男女字符串离散成 0 和 1
df.loc[df["Sex"] == "male", "Sex"] = 0
df.loc[df["Sex"] == "female", "Sex"] = 1
# 将空缺的登岸港口设为最多的港口
df['Embarked'] = df['Embarked'].fillna( df['Embarked'].value_counts().index[0])
# 将登岸港口离散化为 0、1、2
df.loc[df["Embarked"] == "S", "Embarked"] = 0
df.loc[df["Embarked"] == "C", "Embarked"] = 1
df.loc[df["Embarked"] == "Q", "Embarked"] = 2
# 输出检验一下
print(df)

# 随机森林预测数据
rfc = RandomForestClassifier(n_estimators = 12, random_state = 6, n_jobs = -1)
```

```
rfc = rfc.fit(X, Y)
pred = rfc.predict(df)
# 格式化预测数据
pred = pd.DataFrame({'PassengerId':ID.values, 'Survived':pred})
# 导出预测数据
pred.to_csv(r'D:\作业\数据分析\out.csv',index=None)

# 相关系数分析
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['font.family']='sans-serif'
plt.rcParams['axes.unicode_minus'] = False
# fig = plt.figure()
# fig.set(alpha=0.2) # 设定图表颜色 alpha 参数
# 可显示 1000 行
pd.set_option("display.max_columns", 1000)
# 导入原数据文件
iris = pd.read_csv(r'D:\作业\数据分析\train.csv',index_col='PassengerId')
# 查看一下
df = pd.read_csv(r'D:\作业\数据分析\train.csv', names=['乘客 ID','是否幸存','仓位等级','姓名','性别','年龄','兄弟姐妹个数','父母子女个数','船票信息','票价','客舱','登船港口'],index_col='乘客 ID',header=0)
print(df.head(5))

# 数据预处理
# 将年龄空的设为年龄中位数
iris["Age"] = iris["Age"].fillna(iris["Age"].median())

# 将空缺的性别设为男
iris["Sex"] = iris["Sex"].fillna("male")
# 将男女字符串离散成 0 和 1
iris.loc[iris["Sex"] == "male", "Sex"] = 0
iris.loc[iris["Sex"] == "female", "Sex"] = 1

# 将票价空缺的填为平均值
iris['Fare'] = iris['Fare'].fillna(iris['Fare'].mean())

# 将空缺的登岸港口设为 S
iris["Embarked"] = iris["Embarked"].fillna('S')
# 将登岸港口离散化为 0、1、2
iris.loc[iris["Embarked"] == "S", "Embarked"] = 0
iris.loc[iris["Embarked"] == "C", "Embarked"] = 1
iris.loc[iris["Embarked"] == "Q", "Embarked"] = 2

iris["Sex"] = iris["Sex"].astype(int)
```

```
x = iris.corr().index
y = round(iris.corr().iloc[0,:],4)

fig, ax = plt.subplots(figsize=(10, 7))
ax.bar(
    x = x,  # Matplotlib 自动将非数值变量转化为 x 轴坐标
    height = y,  # 柱子高度，y 轴坐标
    width = 0.6,  # 柱子宽度，默认 0.8，两根柱子中心的距离默认为 1.0
    align = "center",  # 柱子的对齐方式，'center' or 'edge'
    color = "yellow",  # 柱子颜色
    edgecolor = "orange",  # 柱子边框的颜色
    linewidth = 1.5  # 柱子边框线的大小
)

ax.set_title(u"相关性系数", fontsize = 18)

xticks = ax.get_xticks()
for i in range(len(y)):
    xy = (xticks[i], y[i] * 1.03)
    s = str(y[i])
    ax.annotate(
        s = s,  # 要添加的文本
        xy = xy,  # 将文本添加到哪个位置
        fontsize = 12,  # 标签大小
        color = "blue",  # 标签颜色
        ha = "center",  # 水平对齐
        va = "baseline"  # 垂直对齐
    )
plt.show()

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from pandas import Series,DataFrame
taitan = pd.read_csv(r'D:\作业\数据分析\train.csv')

plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['font.family']='sans-serif'
plt.rcParams['axes.unicode_minus'] = False
fig = plt.figure()
fig.set(alpha=0.2)

plt.subplot2grid((2,3),(0,0))
taitan.Survived.value_counts().plot(kind='bar')
plt.title(u"表 A 获救情况 (1 为获救)")
```

```
plt.ylabel(u"人数")

plt.subplot2grid((2,3),(0,1))
taitan.Pclass.value_counts().plot(kind="bar")
plt.ylabel(u"人数")
plt.title(u"表 B 乘客船舱号分布")

plt.subplot2grid((2,3),(0,2))
plt.scatter(taitan.Survived, taitan.Age)
plt.ylabel(u"年龄")
plt.grid(b=True, which='major', axis='y')
plt.title(u"表 C 按年龄看获救分布 (1 为获救)")

plt.subplot2grid((2,3),(1,0), colspan=2)
taitan.Age[taitan.Pclass == 1].plot(kind='kde')
taitan.Age[taitan.Pclass == 2].plot(kind='kde')
taitan.Age[taitan.Pclass == 3].plot(kind='kde')
plt.xlabel(u"年龄")
plt.ylabel(u"密度")
plt.title(u"表 D 各船舱号的乘客年龄分布")
plt.legend((u'头等舱', u'2 等舱',u'3 等舱'),loc='best')

plt.subplot2grid((2,3),(1,2))
taitan.Embarked.value_counts().plot(kind='bar')
plt.title(u"表 E 各登船口岸上船人数")
plt.ylabel(u"人数")
# plt.show()

S0 = taitan.Pclass[taitan.Survived == 0].value_counts()
S1 = taitan.Pclass[taitan.Survived == 1].value_counts()
df = pd.DataFrame({u'获救':S1,u'未获救':S0})
df.plot(kind = 'bar', stacked = True)
plt.title(u'各船舱号乘客的获救情况')
plt.xlabel(u'船舱号')
plt.ylabel(u'人数')
plt.show()

Sm = taitan.Survived[taitan.Sex == 'male'].value_counts()
Sf = taitan.Survived[taitan.Sex == 'female'].value_counts()
df = pd.DataFrame({u'未获救':Sm,u'获救':Sf})
df.plot(kind = 'bar', stacked = True)
plt.title(u'按性别看获救情况')
plt.xlabel(u'性别')
plt.ylabel(u'人数')
plt.show()
```

```
fig = plt.figure()
plt.title(u'根据船舱号和性别的获救情况')

ax1 = fig.add_subplot(141)
taitan.Survived[taitan.Sex == 'female'][taitan.Pclass != 3].value_counts().plot(kind =
'bar', label = 'female high class', color = '#FA2479')
ax1.set_xticklabels([u'获救',u'未获救'], rotation = 0)
ax1.legend([u'女性/高级舱'], loc = 'best')

ax2 = fig.add_subplot(142, sharey = ax1)
taitan.Survived[taitan.Sex == 'female'][taitan.Pclass == 3].value_counts().plot(kind =
'bar', label = 'female low class', color = 'pink')
ax2.set_xticklabels([u"未获救", u"获救"], rotation=0)
plt.legend([u"女性/低级舱"], loc='best')

ax3 = fig.add_subplot(143, sharey = ax1)
taitan.Survived[taitan.Sex == 'male'][taitan.Pclass != 3].value_counts().plot(kind =
'bar', label = 'male high class', color = 'lightblue')
ax3.set_xticklabels([u'未获救',u'获救'], rotation = 0)
plt.legend([u'男性/高级舱'], loc = 'best')

ax4 = fig.add_subplot(144, sharey = ax1)
taitan.Survived[taitan.Sex == 'male'][taitan.Pclass == 3].value_counts().plot(kind =
'bar', label = 'male low class', color = 'steelblue')
ax4.set_xticklabels([u'未获救',u'获救'], rotation = 0)
plt.legend([u'男性/低级舱'], loc = 'best')
plt.show()

fig = plt.figure()
fig.set(alpha = 0.2)
S0 = taitan.Embarked[taitan.Survived == 0].value_counts()
S1 = taitan.Embarked[taitan.Survived == 1].value_counts()
df = pd.DataFrame({u'获救':S1,u'未获救':S0})
df.plot(kind = 'bar', stacked = True)
plt.title(u'各登陆港口乘客的获救情况')
plt.xlabel(u'登陆港口')
plt.ylabel(u'人数')
plt.show()
```

总结：

（针对个人项目的完成情况进行学术上的总结，针对大数据分析课程学习的所得、期待、不足、建议、问题进行个人总结）

个人项目总结：

1. 数据再分析

   从相关性系数可以得知，是否存活这一属性与性别、船舱的等级、票价有很大的相关性，**女性、船舱的等级高、票价高的乘客有更大的概率存活下来，而男性、船舱等级低、票价便宜的乘客存活的概率很低。**

从对其他的属性的分析还可以发现，**船舱号的等级越低，票价便宜，年龄密度的峰值越小(即年轻人越多)，说明年轻的人更喜欢买平价票或者年轻人的资本积累少；在 S 口上岸的人最多，此船有很大可能是从 S 口出发的；虽然在不同的船舱的初始人数不同，但是最后存活的人数却差距不大，可以推测在不同的船舱救生艇的数量几乎也差距不多；即便男性初始数量比女性要多一半，但男性的存活率却很低，而女性的存活率却高达 70%；在高级舱的女性存活概率极高，而在低级舱的女性存活比例为 0.5；无论是高级舱还是低级舱的男性未存活的概率都远大于存活概率，低级舱内的男性未存活概率更是极高。**

2. 数据总结

从相关系数分析可以看出，性别是最主要因素而且年龄小的有更高的存活率。妇女优先，儿童优先、泰坦尼克号再现英国人几千年的骑士精神！影响了一代又一代人，向英国骑士精神致敬！

3. 算法总结

在对数据进行处理后，对随机森林算法进行了参数优化，并使用随机森林算法成功的对数据进行了预测分析、交叉验证，而且进行了跑分而且又重新取了跑分最高的算法参数进行了最终的预测。

4. 项目不足

掌握的数据分析算法太少，没有使用一些其他的数据分析算法进行实现(如线性回归等);对于 plt 的使用还不够熟练，对 python 的使用也不够熟悉导致我不能完成对所有数据的相关性的图示分析。

学期学习所得：

通过这学期的大数据分析的课程学习，我学会了 python 中如何对文件进行读取和处理，对 numpy、pandas、plt 的理解和使用更加熟练，学会了很多数据分析的方法，参加的很多数据分析的具体案例，这些极大的增长了我对数据分析学习的经验和兴趣。

课程建议：

希望可以老师带队对最新的 kaggle 比赛进行实时的数据分析讲解，从现实项目中学习。

参考文献：

1. 赵娟，基于信息流动力学的通信网络性能可靠性建模与分析，通信学报，01 卷，08 期，2011 年
2. 纪春芳，数据分析在数据业务监控中的应用，计量学报，04 卷，10 期，2011 年
3. 古喜庆，基于数据仓库和数据挖掘技术的客户数据分析系统设计与实现，北京邮电大学学报，02 卷，05 期，2007 年

备注：无

1. 文件名命名方式：学号+姓名+题目.pdf，例如：“2020010101+张三+星巴克店面位置分析.pdf”，文件名命名错误者，不予统计成绩。
2. 注意文档整体格式，按照相应模块的字体、字号进行书写，标题宋体小四号加粗，正文宋体五号，代码等宽英文字体 11 号，参考文献：姓名，文章名，期刊名，卷，期，年。
3. 评分标准：题目选题（10 分），数据集采集、清洗、整理（20 分），分析观点（准确性 20 分、充分度 10 分、创新性 10 分），方法描述（10 分），代码（5 分），总结（10 分），参考文献（5 分）。