



人脸识别!



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

转 Kaggle泰坦尼克预测(完整分析)

2017年06月26日 15:53:26

1.引言

先说一句，年末双十一什么的一来，真是非(mang)常(cheng)欢(gou)乐(le)！然后push自己抽出时间来写这篇blog的原因也非常简单

- 写完前两篇逻辑回归的介绍和各个角度理解之后，我们讨论群(戳我入群)的小伙伴们纷纷表示『好像很高级的样纸，but 然并卵 啊！你们倒是拿点东西！么！用！啊！』
- talk is cheap, show me the code !
- no example say a jb !

OK，OK，这就来了咯，同学们别着急，我们先找个简单的实际例子，来看看，所谓的**数据挖掘**或者**机器学习实际应用**到底是怎么样一

『喂，那几个说要**看大数据**上机器学习应用的，对，就是说你们！别着急好么，我们之后拉点大一点实际数据用liblinear或者spark,ML个实例入门嘛』

好了，我是一个严肃的技术研究和分享者，咳咳，不能废话了，各位同学继续往下看吧！

2.背景

2.1 关于Kaggle


- 我是Kaggle地址，翻我牌子
- 亲，逼格这么高的地方，你一定听过对不对？是！这就是那个无数『数据挖掘先驱』们，在回答“枪我有了，哪能找到靶子练手啊？”时候的答案！
- 这是一个要数据有数据，要实际应用场景有场景，要一起在数据挖掘领域high得不要不要的小伙伴就有小伙伴的地方啊！！！！

艾玛，逗逼模式开太猛了。恩，不闹，不闹，说正事，Kaggle是一个数据分析建模的应用竞赛平台，有点类似KDD-CUP（国际知识发现和数据挖掘研究者可以将问题背景、数据、期望指标等发布到Kaggle上，以竞赛的形式向广大的数据科学家征集解决方案。而热爱数(dong)据(shou)挖(zhe)掘(载/分析数据，使用统计/机器学习/数据挖掘等知识，建立**算法**模型，得出结果并提交，排名top的可能会有奖金哦！

2.2 关于泰坦尼克号之灾

- 带大家去该问题页面溜达一圈吧

- 下面是问题背景页



Knowledge · 3,412 teams

Titanic: Machine Learning from Disaster

Fri 28 Sep 2012Thu 31 Dec 2015 (54 days to go)

Dashboard

Home

Data

Make a submission

Information

Forum

Scripts

Competition Details » Get the Data » Make a submission

Predict survival on the Titanic using Excel, Python, R & Random Forests


See best practice code and explore visualizations of the Titanic dataset on Kaggle Scripts. Submit directly to the competition, no data download or local environment needed!

The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.

In this challenge, we ask you to complete the analysis of what sorts of people were likely to

- 下面是可下载Data的页面



Knowledge · 3,411 teams

Titanic: Machine Learning from Disaster

Fri 28 Sep 2012Thu 31 Dec 2015 (54 days to go)

Dashboard

Home

Data

Make a submission

Information

Forum


Scripts

Competition Details » Get the Data » Make a submission

Data Files

File Name	Available Formats
train	.csv (59.76 kb)
gendermodel	.csv (3.18 kb)
genderclassmodel	.csv (3.18 kb)
test	.csv (27.96 kb)
gendermodel	.py (3.58 kb)
genderclassmodel	.py (5.63 kb)
myfirstforest	.py (3.99 kb)

- 下面是小伙伴们最爱的forum页面，你会看到各种神级人物厉(qi)害(pa)的数据处理/建模想法，你会直视『世界真奇妙』。



Knowledge · 3,410 teams

Titanic: Machine Learning from Disaster

Fri 28 Sep 2012Thu 31 Dec 2015 (54 days to go)

Dashboard

Competition Forum

New topicStart Watching

Search

12345678910...2122

Votes	432 topics, 1,777 posts	Replies	Views	Last Post
20	Revised (and added) Tutorials by Ramzi R, 19 months ago	13	12614	hehe 2 months ago
23	Rolling Leaderboards by William Cukierski, 2 years ago	10	6781	John Uckele 16 months ago
0	Looking for someone to team up with by JeelShah, 10 days ago	2	147	grazim 18 hours ago



人脸识别!

联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

- 就是那个大家都熟悉的『Jack and Rose』的故事，豪华游艇倒了，大家都惊恐逃生，可是救生艇的数量有限，无法人人都得救！』，所以是否获救其实并非随机，而是基于一些背景有rank先后的。
- 训练和测试数据是一些乘客的个人信息以及存活状况，要尝试根据它生成合适的模型并预测其他人的存活状况。
- 对，这是一个二分类问题，是我们之前讨论的logistic regression所能处理的范畴。

3.说明

接触过Kaggle的同学们可能知道这个问题，也可能知道RandomForest和SVM等等算法，甚至还对多个模型做过融合，取得过非常不错的成绩。如果你对这个问题感兴趣，你可以自行略过。

我们因为之前只介绍了Logistic Regression这一种分类算法。所以本次的问题解决过程和优化思路，都集中在这种算法上。其余的方法，我们就不细说了。说点个人的观点。不一定正确。

『解决一个问题的方法和思路不止一种』
『没有所谓的机器学习算法优劣，也没有绝对高性能的机器学习算法，只有在特定的场景、数据和特征下更合适的机器学习算法。』

4.怎么做？

手把手教程马上就来，先来两条我看到的，觉得很重要的经验。

1. 印象中Andrew Ng老师似乎在coursera上说过，应用机器学习，千万不要一上来就试图做到完美，先撸一个baseline的model，然后逐步提高，所谓后续步骤可能包括『分析model现在的状态(欠/过拟合)，分析我们使用的feature的作用大小，进行feature selection，找出哪些feature是重要的，哪些是不重要的，然后进行feature engineering，产生新的feature』等等。
2. Kaggle上的大神们，也分享过一些experience，说几条我记得的哈：
 1. 『对数据的认识太重要了！』
 2. 『数据中的特殊点/离群点的分析和处理太重要了！』
 3. 『特征工程(feature engineering)太重要了！在很多Kaggle的场景下，甚至比model本身还要重要』
 4. 『要做模型融合(model ensemble)啊啊啊！』

更多的经验分享请加讨论群，具体方式请联系作者，或者参见《“ML学分计划”说明书》

5.初探数据

先看看我们的数据，长什么样吧。在Data下我们train.csv和test.csv两个文件，分别存着官方给的训练和测试数据。

```
import pandas as pd #数据分析
import numpy as np  #科学计算
from pandas import Series,DataFrame

data_train = pd.read_csv("/Users/Hanxiaoyang/Titanic_data/Train.csv")
data_train
```

pandas是常用的Python数据处理包，把csv文件读入成dataframe格式，我们在ipython notebook中，看到data_train如下所示：

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cummings, Mrs. John Bradley (Florence Briggs Th...	female	38	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2	3	1	349909	21.0750	NaN	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14	1	0	237736	30.0708	NaN	C

这就是典型的数据frame格式，如果你没接触过这种格式，完全没有关系，你就把它想象成Excel里面的列好了。我们看到，总共有12列，其中Survived字段表示的是该乘客是否获救，其余都是乘客的个人信息，包括：



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网网
北京互联网



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

- Name => 乘客姓名
- Sex => 性别
- Age => 年龄
- SibSp => 堂兄弟/妹个数
- Parch => 父母与小孩个数
- Ticket => 船票信息
- Fare => 票价
- Cabin => 客舱
- Embarked => 登船港口

逐条往下看，要看完这么多条，眼睛都有一种要瞎的赶脚。好吧，我们让dataframe自己告诉我们一些信息，如下所示：

```
data_train.info()
```

1

看到了如下的信息：

```
1]: data_train.info()  
#我们发现有一些列，比如说Cabin，有非常多的缺失值  
#另外一些我们感觉在此场景中会有影响的属性，比如Age，也有一些缺失值  
  
<class 'pandas.core.frame.DataFrame'>  
Int64Index: 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: 90.5+ KB
```

上面的数据说啥了？它告诉我们，训练数据中总共有891名乘客，但是很不幸，我们有些属性的数据不全，比如说：

- Age（年龄）属性只有714名乘客有记录
- Cabin（客舱）更是只有204名乘客是已知的

似乎信息略少啊，想再瞄一眼具体数据数值情况呢？恩，我们用下列的方法，得到数值型数据的一些分布(因为有些属性，比如姓名，是文本型；而船港口，是类目型。这些我们用下面的函数是看不到的)：

data_train.describe()							
	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

我们从上面看到更进一步的什么信息呢？
mean字段告诉我们，大概0.383838的人最后获救了，2/3等舱的人数比1等舱要多，平均乘客年龄大概是29.7岁(计算这个时候会略掉无记录的)等等

6.数据初步分析

每个乘客都这么多属性，那我们咋知道哪些属性更有用，而又应该怎么用它们啊？说实话这会儿我也不知道，但我们记得前面提到过

- 『对数据的认识太重要了！』

重要的事情说三遍，恩，说完了。仅仅最上面的对数据了解，依旧无法给我们提供想法和思路。我们再深入一点来看看我们的数据，**d**之间有着什么样的关系呢。

6.1 乘客各属性分布

脑容量太有限了...数值看花眼了。我们还是统计统计，画些图来看看属性和结果之间的关系好了，代码如下：

```
import matplotlib.pyplot as plt
fig = plt.figure()
fig.set(alpha=0.2) # 设定图表颜色alpha参数

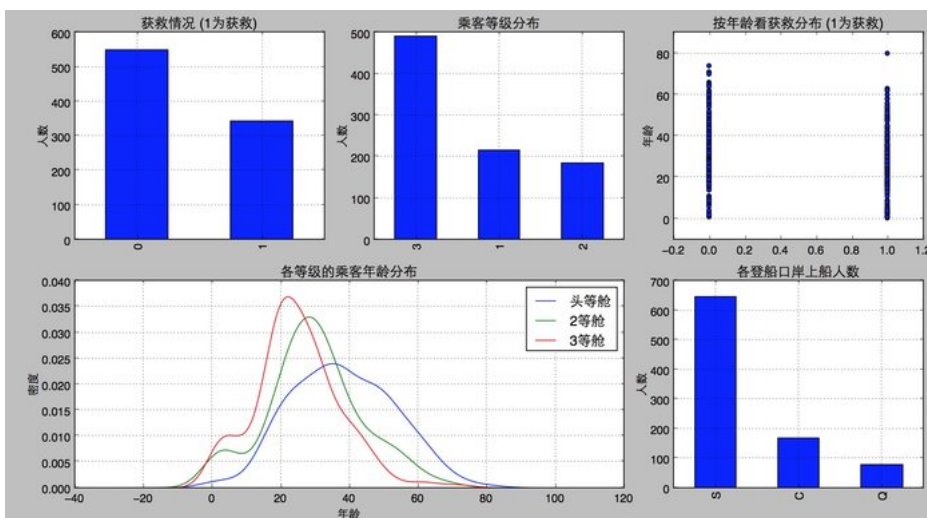
plt.subplot2grid((2,3),(0,0)) # 在一张大图里分列几个小图
data_train.Survived.value_counts().plot(kind='bar') # 柱状图
plt.title(u"获救情况 (1为获救)") # 标题
plt.ylabel(u"人数")

plt.subplot2grid((2,3),(0,1))
data_train.Pclass.value_counts().plot(kind="bar")
plt.ylabel(u"人数")
plt.title(u"乘客等级分布")

plt.subplot2grid((2,3),(0,2))
plt.scatter(data_train.Survived, data_train.Age)
plt.ylabel(u"年龄") # 设定纵坐标名称
plt.grid(b=True, which='major', axis='y')
plt.title(u"按年龄看获救分布 (1为获救)")

plt.subplot2grid((2,3),(1,0), colspan=2)
data_train.Age[data_train.Pclass == 1].plot(kind='kde')
data_train.Age[data_train.Pclass == 2].plot(kind='kde')
data_train.Age[data_train.Pclass == 3].plot(kind='kde')
plt.xlabel(u"年龄") # plots an axis lable
plt.ylabel(u"密度")
plt.title(u"各等级的乘客年龄分布")
plt.legend((u'头等舱', u'2等舱', u'3等舱'), loc='best') # sets our legend for our graph.

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



bingo，图还是比数字好看多了。所以我们在图上可以看出来，被救的人300多点，不到半数；3等舱乘客灰常多；遇难和获救的人年龄似乎跨度都总体趋势似乎也一致，2/3等舱乘客20岁多点的人最多，1等舱40岁左右的最多(→_→似乎符合财富和年龄的分配哈，咳咳，别理我，我瞎扯的) C、Q递减，而且S远多于另外俩港口。

这个时候我们可能会有一些想法了：

加入CSDN，享受更精准的内容推荐，与500万程序员共同成长！

登录

注册

×



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

- 年龄对获救概率也一定是有影响的，毕竟前面说了，副船长还说『小孩和女士先走』呢
- 和登船港口是不是有关系呢？也许登船港口不同，人的出身地位不同？

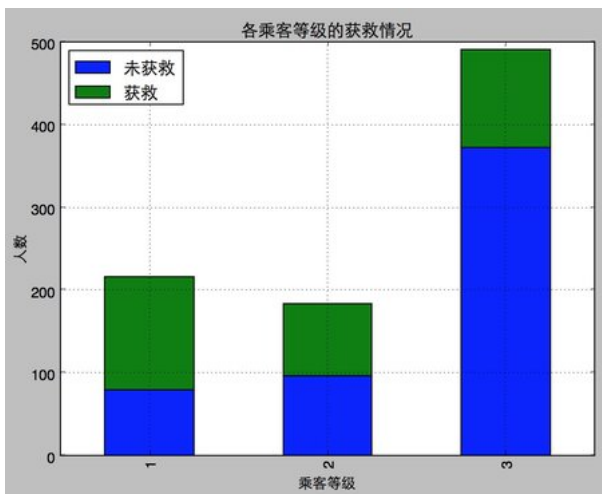
口说无凭，空想无益。老老实实再来统计统计，看看这些属性值的统计分布吧。

6.2 属性与获救结果的关联统计

#看看各乘客等级的获救情况

```
fig = plt.figure()
fig.set(alpha=0.2) # 设定图表颜色alpha参数
```

```
Survived_0 = data_train.Pclass[data_train.Survived == 0].value_counts()
Survived_1 = data_train.Pclass[data_train.Survived == 1].value_counts()
df=pd.DataFrame({u'获救':Survived_1, u'未获救':Survived_0})
df.plot(kind='bar', stacked=True)
plt.title(u"各乘客等级的获救情况")
plt.xlabel(u"乘客等级")
plt.ylabel(u"人数")
plt.show()
```



啧啧，果然，钱和地位对舱位有影响，进而对获救的可能性也有影响啊←_←

咳咳，跑题了，我想说的是，明显等级为1的乘客，获救的概率高很多。恩，这个一定是影响最后获救结果的一个特征。

#看看各性别的获救情况

```
fig = plt.figure()
fig.set(alpha=0.2) # 设定图表颜色alpha参数
```

```
Survived_m = data_train.Survived[data_train.Sex == 'male'].value_counts()
Survived_f = data_train.Survived[data_train.Sex == 'female'].value_counts()
df=pd.DataFrame({u'男性':Survived_m, u'女性':Survived_f})
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()
```

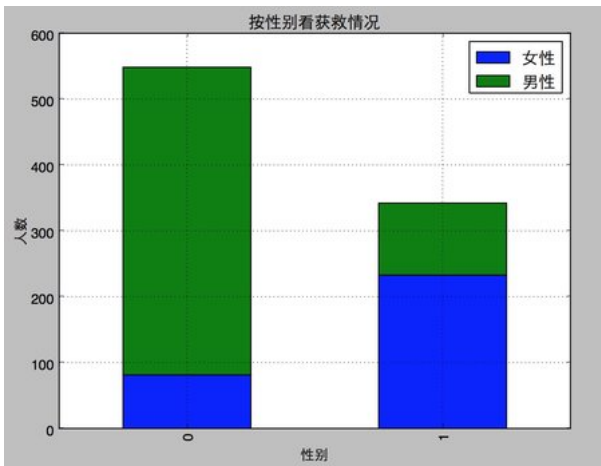


联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网



歪果盆友果然很尊重lady，lady first践行得不错。性别无疑也要作为重要特征加入最后的模型之中。

再来个详细版的好了。

```
#然后我们再来看看各种舱级别情况下各性别的获救情况
fig=plt.figure()
fig.set(alpha=0.65) # 设置图像透明度，无所谓
plt.title(u"根据舱等级和性别的获救情况")

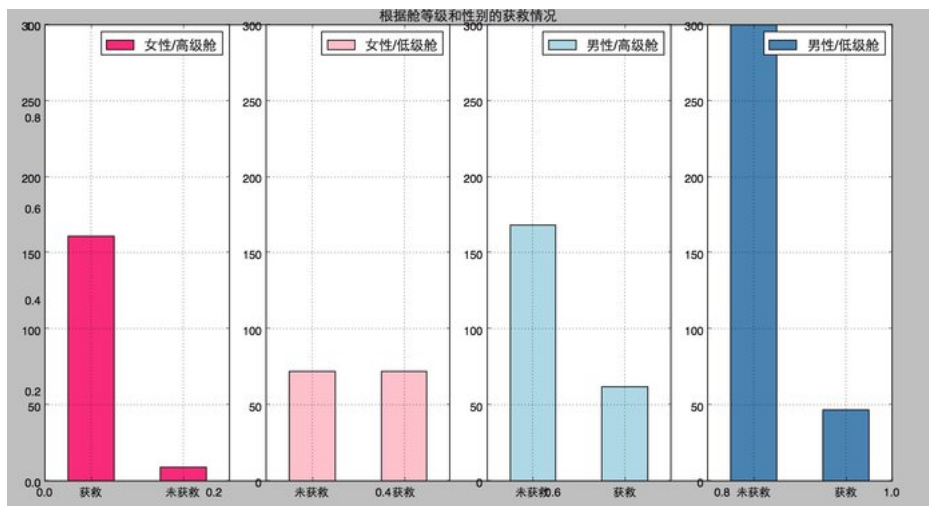
ax1=fig.add_subplot(141)
data_train.Survived[data_train.Sex == 'female'][data_train.Pclass != 3].value_counts().plot(kind='bar', label="female, high class")
ax1.set_xticklabels([u"获救", u"未获救"], rotation=0)
ax1.legend([u"女性/高级舱"], loc='best')

ax2=fig.add_subplot(142, sharey=ax1)
data_train.Survived[data_train.Sex == 'female'][data_train.Pclass == 3].value_counts().plot(kind='bar', label='female, low class')
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)
data_train.Survived[data_train.Sex == 'male'][data_train.Pclass != 3].value_counts().plot(kind='bar', label='male, high class')
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)
data_train.Survived[data_train.Sex == 'male'][data_train.Pclass == 3].value_counts().plot(kind='bar', label='male low class')
ax4.set_xticklabels([u"未获救", u"获救"], rotation=0)
plt.legend([u"男性/低级舱"], loc='best')

plt.show()
```



恩，坚定了之前的判断。

加入CSDN，享受更精准的内容推荐，与500万程序员共同成长！

登录

注册

×



联系我们



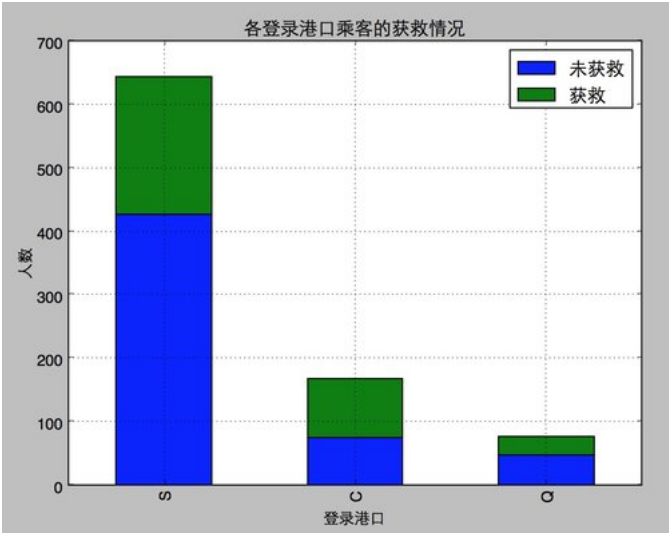
关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

```
fig = plt.figure()
fig.set(alpha=0.2) # 设定图表颜色alpha参数

Survived_0 = data_train.Embarked[data_train.Survived == 0].value_counts()
Survived_1 = data_train.Embarked[data_train.Survived == 1].value_counts()
df=pd.DataFrame({'u'获救':Survived_1, u'未获救':Survived_0})
df.plot(kind='bar', stacked=True)
plt.title(u"各登录港口乘客的获救情况")
plt.xlabel(u"登录港口")
plt.ylabel(u"人数")

plt.show()
```



下面我们来看看 堂兄弟/妹，孩子/父母有几人，对是否获救的影响。

```
g = data_train.groupby(['SibSp', 'Survived'])
df = pd.DataFrame(g.count()['PassengerId'])
print df

g = data_train.groupby(['SibSp', 'Survived'])
df = pd.DataFrame(g.count()['PassengerId'])
print df
```

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9



联系我们



关于 招聘
©2018 CSDN
百度提供
经营性网站备案
网络110报警
中国互联网
北京互联网

		PassengerId
SibSp	Survived	
0	0	398
	1	210
1	0	97
	1	112
2	0	15
	1	13
3	0	12
	1	4
4	0	15
	1	3
5	0	5
8	0	7

		PassengerId
Parch	Survived	
0	0	445
	1	233
1	0	53
	1	65
2	0	40
	1	40
3	0	2
	1	3
4	0	4
5	0	4
	1	1
6	0	1

好吧，没看出特别特别明显的规律(为自己的智商感到捉急...), 先作为备选特征，放一放。

```
#ticket是船票编号，应该是unique的，和最后的结果没有太大的关系，先不纳入考虑的特征范畴把
#cabin只有204个乘客有值，我们先看看它的一个分布
data_train.Cabin.value_counts()
```

1
2
3
4
5



人脸识别!



联系我们



关于 招聘

©2018 CSDN

 百度提供

经营性网站备案

网络110报警

中国互联网

北京互联网



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

部分结果如下：

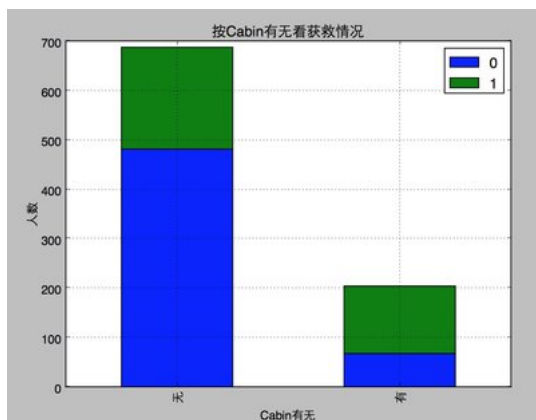
C23 C25 C27	4
G6	4
B96 B98	4
D	3
C22 C26	3
E101	3
F2	3
F33	3
B57 B59 B63 B66	2
C68	2
B58 B60	2
E121	2
D20	2
E8	2
E44	2
B77	2
C65	2
D26	2
E24	2
E25	2
B20	2
C93	2
D33	2
E67	2
D35	2
D36	2
C52	2
F4	2
C125	2
C124	2

这三三两两的...如此不集中...我们猜一下，也许，前面的ABCDE是指的甲板位置、然后编号是房间号？...好吧，我瞎说的，别当真...

关键是Cabin这鬼属性，应该算作类目型的，本来缺失值就多，还如此不集中，注定是个棘手货...第一感觉，这玩意儿如果直接按照类目特征处理的因子化后的特征都拿不到什么权重。加上有那么多缺失值，要不我们先把Cabin缺失与否作为条件(虽然这部分信息缺失可能并非未登记，maybe这样做未必妥当)，先在有无Cabin信息这个粗粒度上看看Survived的情况好了。

```
fig = plt.figure()
fig.set(alpha=0.2) # 设定图表颜色alpha参数

Survived_cabin = data_train.Survived[pd.notnull(data_train.Cabin)].value_counts()
Survived_nocabin = data_train.Survived[pd.isnull(data_train.Cabin)].value_counts()
df=pd.DataFrame({u'有':Survived_cabin, u'无':Survived_nocabin}).transpose()
df.plot(kind='bar', stacked=True)
plt.title(u"按Cabin有无看获救情况")
plt.xlabel(u"Cabin有无")
plt.ylabel(u"人数")
plt.show()
```



咳咳，有Cabin记录的似乎获救概率稍高一些，先这么着放一放吧。

大体数据的情况看了一遍，对感兴趣的属性也有个大概的了解了。
下一步干啥？咱们该处理处理这些数据，为机器学习建模做点准备了。

对了，我这里说的数据预处理，其实就包括了很多Kaggler津津乐道的feature engineering过程，灰常灰常有必要！

『特征工程(feature engineering)太重要了！』
『特征工程(feature engineering)太重要了！』
『特征工程(feature engineering)太重要了！』

恩，重要的事情说三遍。

先从最突出的数据属性开始吧，对，Cabin和Age，有丢失数据实在是下一步工作影响太大。

先说Cabin，暂时我们就按照刚才说的，按Cabin有无数据，将这个属性处理成Yes和No两种类型吧。

再说Age：

通常遇到缺值的情况，我们会有几种常见的处理方式

- 如果缺值的样本占总数比例极高，我们可能就直接舍弃了，作为特征加入的话，可能反倒带入noise，影响最后的结果了
- 如果缺值的样本适中，而该属性为非连续值特征属性(比如说类目属性)，那就把NaN作为一个新类别，加到类别特征中
- 如果缺值的样本适中，而该属性为连续值特征属性，有时候我们会考虑给定一个step(比如这里的age，我们可以考虑每隔2/3岁为一个步长)，然后归入属性类目中。
- 有些情况下，缺失的值个数并不是特别多，那我们也可以试着根据已有的值，拟合一下数据，补充上。

本例中，后两种处理方式应该都是可行的，我们先试试拟合补全吧(虽然说没有特别多的背景可供我们拟合，这不一定是一个多么好的)

我们这里用scikit-learn中的RandomForest来拟合一下缺失的年龄数据(注：RandomForest是一个用在原始数据中做不同采样，建立多颗Decision Tree等等来降低过拟合现象，提高结果的机器学习算法，我们之后会介绍到)

```
from sklearn.ensemble import RandomForestRegressor

### 使用 RandomForestClassifier 填补缺失的年龄属性
def set_missing_ages(df):

    # 把已有的数值型特征取出来丢进Random Forest Regressor中
    age_df = df[['Age', 'Fare', 'Parch', 'SibSp', 'Pclass']]

    # 乘客分成已知年龄和未知年龄两部分
    known_age = age_df[age_df.Age.notnull()].as_matrix()
    unknown_age = age_df[age_df.Age.isnull()].as_matrix()

    # y即目标年龄
    y = known_age[:, 0]

    # X即特征属性值
    X = known_age[:, 1:]

    # fit到RandomForestRegressor之中
    rfr = RandomForestRegressor(random_state=0, n_estimators=2000, n_jobs=-1)
    rfr.fit(X, y)

    # 用得到的模型进行未知年龄结果预测
    predictedAges = rfr.predict(unknown_age[:, 1::])

    # 用得到的预测结果填补原缺失数据
    df.loc[ (df.Age.isnull()), 'Age' ] = predictedAges

    return df, rfr

def set_Cabin_type(df):
    df.loc[ (df.Cabin.notnull()), 'Cabin' ] = "Yes"
    df.loc[ (df.Cabin.isnull()), 'Cabin' ] = "No"
    return df

data_train, rfr = set_missing_ages(data_train)
```



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.000000	1	0	A/5 21171	7.2500	No	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.000000	1	0	PC 17599	71.2833	Yes	C
2	3	1	3	Heikinen, Miss. Laina	female	26.000000	0	0	STON/O2. 3101282	7.9250	No	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.000000	1	0	113803	53.1000	Yes	S
4	5	0	3	Allen, Mr. William Henry	male	35.000000	0	0	373450	8.0500	No	S
5	6	0	3	Moran, Mr. James	male	23.828953	0	0	330877	8.4583	No	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.000000	0	0	17463	51.8625	Yes	S
7	8	0	3	Paisson, Master. Gosta Leonard	male	2.000000	3	1	349909	21.0750	No	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.000000	0	2	347742	11.1333	No	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.000000	1	0	237736	30.0708	No	C
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.000000	1	1	PP 9549	16.7000	Yes	S

恩。目的达到，OK了。

因为逻辑回归建模时，需要输入的特征都是数值型特征，我们通常会先对类目型的特征因子化。

什么叫做因子化呢？举个例子：

以Cabin为例，原本一个属性维度，因为其取值可以是['yes' , ' no']，而将其平展开为 ' Cabin_yes' , ' Cabin_no' 两个属性

- 原本Cabin取值为yes的，在此处的“ Cabin_yes” 下取值为1，在“ Cabin_no” 下取值为0
- 原本Cabin取值为no的，在此处的“ Cabin_yes” 下取值为0，在“ Cabin_no” 下取值为1

我们使用pandas的“ get_dummies” 来完成这个工作，并拼接在原来的“ data_train” 之上，如下所示。

```
dummies_Cabin = pd.get_dummies(data_train['Cabin'], prefix= 'Cabin')

dummies_Embarked = pd.get_dummies(data_train['Embarked'], prefix= 'Embarked')

dummies_Sex = pd.get_dummies(data_train['Sex'], prefix= 'Sex')

dummies_Pclass = pd.get_dummies(data_train['Pclass'], prefix= 'Pclass')

df = pd.concat([data_train, dummies_Cabin, dummies_Embarked, dummies_Sex, dummies_Pclass], axis=1)
df.drop(['Pclass', 'Name', 'Sex', 'Ticket', 'Cabin', 'Embarked'], axis=1, inplace=True)
df
```

SibSp	Parch	Fare	Cabin_No	Cabin_Yes	Embarked_C	Embarked_Q	Embarked_S	Sex_female	Sex_male	Pclass_1	Pclass_2	Pclass_3
1	0	7.2500	1	0	0	0	1	0	1	0	0	1
1	0	71.2833	0	1	1	0	0	1	0	1	0	0
0	0	7.9250	1	0	0	0	1	1	0	0	0	1
1	0	53.1000	0	1	0	0	1	1	0	1	0	0
0	0	8.0500	1	0	0	0	1	0	1	0	0	1
0	0	8.4583	1	0	0	1	0	0	1	0	0	1
0	0	51.8625	0	1	0	0	1	0	1	1	0	0
3	1	21.0750	1	0	0	0	1	0	1	0	0	1
0	2	11.1333	1	0	0	0	1	1	0	0	0	1
1	0	30.0708	1	0	1	0	0	1	0	0	1	0

bingo，我们很成功地把这些类目属性全都转成0，1的数值属性了。

这样，看起来，是不是我们需要的属性值都有了，且它们都是数值型属性呢。

有一种临近结果的宠宠欲动感吧，莫急莫急，我们还得做一些处理，仔细看看Age和Fare两个属性，乘客的数值幅度变化，也忒大了吧！！如果大下降的话，会知道，各属性值之间scale差距太大，将对收敛速度造成几万点伤害值！甚至不收敛！（崩溃）...所以我们先用scikit-learn里面的pre货做一个scaling，所谓scaling，其实就是将一些变化幅度较大的特征化到[-1,1]之内。

```
import sklearn.preprocessing as preprocessing
scaler = preprocessing.StandardScaler()
age_scale_param = scaler.fit(df['Age'])
df['Age_scaled'] = scaler.fit_transform(df['Age'], age_scale_param)
```

```
df['Fare_scaled'] = scaler.fit_transform(df['Fare'], fare_scale_param)
df
```

	Cabin_No	Cabin_Yes	Embarked_C	Embarked_Q	Embarked_S	Sex_female	Sex_male	Pclass_1	Pclass_2	Pclass_3	Age_scaled	Fare_scaled
1)	0	0	0	0	1	0	1	0	0	1	-0.561417	-0.502445
	1	1	0	0	1	0	1	0	0	0	0.613177	0.786845
	0	0	0	1	1	0	0	0	1	-0.267768	-0.488854	
2)	1	0	0	1	1	0	1	0	0	0	0.392941	0.420730
	0	0	0	1	0	1	0	0	1	0.392941	-0.486337	
	0	0	1	0	0	1	0	0	1	-0.427149	-0.478116	
3)	1	0	0	1	0	1	1	0	0	0	1.787771	0.395814
	0	0	0	1	0	1	0	0	1	-2.029659	-0.224083	

恩，好看多了，万事俱备，只欠建模。马上就要看到成效了，哈哈。我们把需要的属性值抽出来，转成scikit-learn里面LogisticRegression

8.逻辑回归建模

我们把需要的feature字段取出来，转成numpy格式，使用scikit-learn中的LogisticRegression建模。

```
from sklearn import linear_model

# 用正则取出我们要的属性值
train_df = df.filter(regex='Survived|Age_|SibSp|Parch|Fare_|Cabin_|Embarked_|Sex_|Pclass_.+')
train_np = train_df.as_matrix()

# y即Survival结果
y = train_np[:, 0]

# X即特征属性值
X = train_np[:, 1:]

# fit到RandomForestRegressor之中
clf = linear_model.LogisticRegression(C=1.0, penalty='l1', tol=1e-6)
clf.fit(X, y)

clf
```

good，很顺利，我们得到了一个model，如下：

```
3): LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
    intercept_scaling=1, max_iter=100, multi_class='ovr',
    penalty='l1', random_state=None, solver='liblinear', tol=1e-06,
    verbose=0)
```

先淡定！淡定！你以为把test.csv直接丢进model里就能拿到结果啊...骚年，图样图森破啊！我们的“test_data”也要做和“train_data”一样的

```
data_test = pd.read_csv("/Users/Hanxiaoyang/Titanic_data/test.csv")
data_test.loc[(data_test.Fare.isnull()), 'Fare'] = 0
# 接着我们对test_data做和train_data中一致的特征变换
# 首先用同样的RandomForestRegressor模型填上丢失的年龄
tmp_df = data_test[['Age', 'Fare', 'Parch', 'SibSp', 'Pclass']]
null_age = tmp_df[data_test.Age.isnull()].as_matrix()
# 根据特征属性X预测年龄并补上
X = null_age[:, 1:]
predictedAges = rfr.predict(X)
data_test.loc[(data_test.Age.isnull()), 'Age'] = predictedAges

data_test = set_Cabin_type(data_test)
dummies_Cabin = pd.get_dummies(data_test['Cabin'], prefix='Cabin')
dummies_Embarked = pd.get_dummies(data_test['Embarked'], prefix='Embarked')
dummies_Sex = pd.get_dummies(data_test['Sex'], prefix='Sex')
dummies_Pclass = pd.get_dummies(data_test['Pclass'], prefix='Pclass')

df_test = pd.concat([data_test, dummies_Cabin, dummies_Embarked, dummies_Sex, dummies_Pclass], axis=1)
df_test.drop(['Pclass', 'Name', 'Sex', 'Ticket', 'Cabin', 'Embarked'], axis=1, inplace=True)
df_test['Age_scaled'] = scaler.fit_transform(df_test['Age'], age_scale_param)
df_test['Fare_scaled'] = scaler.fit_transform(df_test['Fare'], fare_scale_param)
```



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

Jabin_No	Cabin_Yes	Embarked_C	Embarked_Q	Embarked_S	Sex_female	Sex_male	Pclass_1	Pclass_2	Pclass_3	Age_scaled	Fare_scaled
	0	0	1	0	0	1	0	0	1	0.307495	-0.496637
	0	0	0	1	1	0	0	0	1	1.256225	-0.511497
	0	0	1	0	0	1	0	1	0	2.394702	-0.463335
	0	0	0	1	0	1	0	0	1	-0.261743	-0.481704
	0	0	0	1	1	0	0	0	1	-0.641235	-0.416740
	0	0	0	1	0	1	0	0	1	-1.248423	-0.471623
	0	0	1	0	1	0	0	0	1	-0.034048	-0.500221
	0	0	0	1	0	1	0	1	0	-0.337642	-0.117238
	0	1	0	0	1	0	0	0	1	-0.944829	-0.507390
	0	0	0	1	0	1	0	0	1	-0.717134	-0.204154
	0	0	0	1	0	1	0	0	1	-0.189852	-0.495444
	0	0	0	1	0	1	1	0	0	1.180327	-0.171000
1	1	0	0	1	1	0	1	0	0	-0.565337	0.837349

不错不错，数据很OK，差最后一步了。
下面就做预测取结果吧！！

```
test = df_test.filter(regex='Age_*|SibSp|Parch|Fare_*|Cabin_*|Embarked_*|Sex_*|Pclass_*')
predictions = clf.predict(test)
result = pd.DataFrame({'PassengerId':data_test['PassengerId'].as_matrix(), 'Survived':predictions.astype(np.int32)})
result.to_csv("/Users/Hanxiaoyang/Titanic_data/logistic_regression_predictions.csv", index=False)
```

	PassengerId	Survived
0	892	0
1	893	0
2	894	0
3	895	0
4	896	1
5	897	0
6	898	1
7	899	0
8	900	1
9	901	0
10	902	0
11	903	0

啧啧，挺好，格式正确，去make a submission啦啦啦！

在Kaggle的Make a submission页面，提交上结果。如下：

2726	new	jeremyyeo	0.76555	1	Mon, 09 Nov 2015 22:45:26
2727	new	JustinL	0.76555	2	Mon, 09 Nov 2015 23:29:01 (-0.3h)
2728	new	NSK	0.76555	10	Tue, 10 Nov 2015 02:05:39 (-1.9h)
2729	new	MichelleNgan	0.76555	2	Tue, 10 Nov 2015 03:18:05
2730	new	Aakash Rana	0.76555	1	Tue, 10 Nov 2015 07:03:38
2731	new	hanxiaoyang	0.76555	3	Tue, 10 Nov 2015 09:39:25
Your Best Entry ↑ You improved on your best score by 0.00957. You just moved up 148 positions on the leaderboard. Tweet this!					
2732	245	sd tin	0.76077	4	Sun, 13 Sep 2015 08:48:27 (-44.4h)
2733	245	ChristopherKeune	0.76077	1	Fri, 11 Sep 2015 13:38:48
2734	245	🏠 🏠 🏠	0.76077	10	Wed, 16 Sep 2015 01:14:37 (-4.1d)
2735	245	MariaShen	0.76077	1	Mon, 14 Sep 2015 03:39:21
2736	245	Mary Nichol	0.76077	4	Wed, 16 Sep 2015 04:18:10 (-18.1h)

0.76555，恩，结果还不错。毕竟，这只是我们简单分析处理过后出的一个baseline模型嘛。

9.逻辑回归系统优化

加入CSDN，享受更精准的内容推荐，与500万程序员共同成长！

登录

注册

×



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

亲，你以为结果提交上了，就完事了？
我不会告诉你，这只是万里长征第一步啊(泪牛满面)！！！这才刚撸完baseline model啊！！！还得优化啊！！！

看过Andrew Ng老师的machine Learning课程的同学们，知道，我们应该分析分析模型现在的状态了，是过/欠拟合？，以确定我们其他操作。我们有一条很著名的learning curves对吧。

不过在现在的场景下，先不着急做这个事情，我们这个baseline系统还有些粗糙，先再挖掘挖掘。

- 首先，Name和Ticket两个属性被我们完整舍弃了(好吧，其实是因为这俩属性，几乎每一条记录都是一个完全不同的值，我们并没
 - 然后，我们想想，年龄的拟合本身也未必是一件非常靠谱的事情，我们依据其余属性，其实并不能很好地拟合预测出未知的年龄
- 友和老人可能得到的照顾会多一些，这样看的话，年龄作为一个连续值，给一个固定的系数，应该和年龄是一个正相关或者负相关情况，所以，说不定我们把年龄离散化，按区间分作类别属性会更合适一些。

上面只是我瞎想的，who knows是不是这么回事呢，老老实实先把得到的model系数和feature关联起来看看。

```
pd.DataFrame({"columns":list(train_df.columns)[1:], "coef":list(clf.coef_.T)})
```

	coef	columns
0	[-0.344189431858]	SibSp
1	[-0.104924350555]	Parch
2	[0.0]	Cabin_No
3	[0.902071479485]	Cabin_Yes
4	[0.0]	Embarked_C
5	[0.0]	Embarked_Q
6	[-0.417226462259]	Embarked_S
7	[1.95649520339]	Sex_female
8	[-0.677484871046]	Sex_male
9	[0.341226064445]	Pclass_1
10	[0.0]	Pclass_2
11	[-1.19410912948]	Pclass_3
12	[-0.523774279397]	Age_scaled
13	[0.0844279740271]	Fare_scaled

首先，大家回去前两篇文章里瞄一眼公式就知道，这些系数为正的特征，和最后结果是一个正相关，反之为负相关。

我们先看看那些权重绝对值非常大的feature，在我们的模型上：

- Sex属性，如果是female会极大提高最后获救的概率，而male会很大程度拉低这个概率。
- Pclass属性，1等舱乘客最后获救的概率会上升，而乘客等级为3会极大地拉低这个概率。
- 有Cabin值会很大程度拉升最后获救概率(这里似乎能看到了一点端倪，事实上从最上面的有无Cabin记录的Survived分布图上看，即使有Cabin记录的乘客也有属性上我们挖掘还不够)
- Age是一个负相关，意味着在我们的模型里，年龄越小，越有获救的优先权(还得回原数据看看这个是否合理)
- 有一个登船港口S会很大程度拉低获救的概率，另外俩港口压根就没啥作用(这个实际上非常奇怪，因为我们从之前的统计图上并没有看到S港口的获救率非常低，用这个feature去掉试试)。
- 船票Fare有小幅度的正相关(并不意味着这个feature作用不大，有可能是我们细化的程度还不够，举个例子，说不定我们得对它离散化，再分至各个乘客等级上?)

噢啦，观察完了，我们现在有一些想法了，但是怎么样才知道，哪些优化的方法是promising的呢？

因为test.csv里面并没有Survived这个字段(好吧，这是废话，这明明就是我们要预测的结果)，我们无法在这份数据上评定我们算法在该场景下的效
而『每做一次调整就make a submission，然后根据结果来判定这次调整的好坏』其实是行不通的...

9.2 交叉验证

重点又来了：

- 『要做交叉验证(cross validation)!』
- 『要做交叉验证(cross validation)!』
- 『要做交叉验证(cross validation)!』

恩，重要的事情说三遍。我们通常情况下，这么做cross validation：把train.csv分成两部分，一部分用于训练我们需要的模型，另外一部分数据果。

人脸识别!

联系我们

关于 招聘

©2018 CSDN

百度提供

经营性网站备案

网络110报警

中国互联网

北京互联网

先简单看看cross validation情况下的打分

```
from sklearn import cross_validation

#简单看看打分情况
clf = linear_model.LogisticRegression(C=1.0, penalty='l1', tol=1e-6)
all_data = df.filter(regex='Survived|Age_.*|SibSp|Parch|Fare_.*|Cabin_.*|Embarked_.*|Sex_.*|Pclass_.*')
X = all_data.as_matrix()[1:]
y = all_data.as_matrix()[0]
print cross_validation.cross_val_score(clf, X, y, cv=5)
```

结果是下面酱紫的：
[0.81564246 0.81005587 0.78651685 0.78651685 0.81355932]

似乎比Kaggle上的结果略高哈，毕竟用的是不是同一份数据集评估的。

等等，既然我们要做交叉验证，那我们干脆先把交叉验证里面的bad case拿出来看看，看看人眼审核，是否能发现什么蛛丝马迹，是被判定错了。再把bad case上得到的想法和前头系数分析的合在一起，然后逐个试试。

下面我们做数据分割，并且在原始数据集上瞄一眼bad case：

```
# 分割数据，按照 训练数据:cv数据 = 7:3的比例
split_train, split_cv = cross_validation.train_test_split(df, test_size=0.3, random_state=0)
train_df = split_train.filter(regex='Survived|Age_.*|SibSp|Parch|Fare_.*|Cabin_.*|Embarked_.*|Sex_.*|Pclass_.*')
# 生成模型
clf = linear_model.LogisticRegression(C=1.0, penalty='l1', tol=1e-6)
clf.fit(train_df.as_matrix()[1:], train_df.as_matrix()[0])

# 对cross validation数据进行预测

cv_df = split_cv.filter(regex='Survived|Age_.*|SibSp|Parch|Fare_.*|Cabin_.*|Embarked_.*|Sex_.*|Pclass_.*')
predictions = clf.predict(cv_df.as_matrix()[1:])

origin_data_train = pd.read_csv("/Users/HanXiaoyang/Titanic_data/Train.csv")
bad_cases = origin_data_train.loc[origin_data_train['PassengerId'].isin(split_cv[predictions != cv_df.as_matrix()[0,0]]['PassengerId'])]
```

我们判定错误的 bad case 中部分数据如下：

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
14	15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14.00	0	0	350406	7.8542	NaN	S
49	50	0	3	Arnold-Franchi, Mrs. Josef (Josefine Franchi)	female	18.00	1	0	349237	17.8000	NaN	S
55	56	1	1	Woolner, Mr. Hugh	male	NaN	0	0	19947	35.5000	C52	S
65	66	1	3	Moubarek, Master. Gerios	male	NaN	1	1	2661	15.2458	NaN	C
68	69	1	3	Andersson, Miss. Erna Alexandra	female	17.00	4	2	3101281	7.9250	NaN	S
85	86	1	3	Backstrom, Mrs. Karl Alfred (Maria Mathilda Gu...	female	33.00	3	0	3101278	15.8500	NaN	S
113	114	0	3	Jussila, Miss. Katriona	female	20.00	1	0	4136	9.8250	NaN	S
140	141	0	3	Boulos, Mrs. Joseph (Sultana)	female	NaN	0	2	2678	15.2458	NaN	C
204	205	1	3	Cohen, Mr. Gurshon "Gus"	male	18.00	0	0	A/5 3540	8.0500	NaN	S
240	241	0	3	Zabour, Miss. Thamine	female	NaN	1	0	2665	14.4542	NaN	C
251	252	0	3	Strom, Mrs. Wilhelm (Eina Matilda Persson)	female	29.00	1	1	347054	10.4625	G6	S

大家可以自己跑一遍试试，拿到bad cases之后，仔细看看。也会有一些猜测和想法。其中会有一部分可能会印证在系数分析部分的猜测，那这些猜高一些。

现在有了“train_df”和“vc_df”两个数据部分，前者用于训练model，后者用于评定和选择模型。可以开始可劲折腾了。

我们随便列一些可能可以做的优化操作：

- Age属性不使用现在的拟合方式，而是根据名称中的『Mr』『Mrs』『Miss』等的平均值进行填充。
- Age不做成一个连续值属性，而是使用一个步长进行离散化，变成离散类目feature。
- Cabin再细化一些，对于有记录的Cabin属性，我们将其分为前面的字母部分(我猜是位置和船层之类的信息)和后面的数字部分(应该是房间号，有意思的事情是，你会发现，这个值大的情况下，似乎获救的可能性高一些)。
- Pclass和Sex俩太重要了，我们试着用它们去组出一个组合属性来试试，这也是另外一种程度的细化。
- 单加一个Child字段，Age<=12的，设为1，其余为0(你去看看数据，确实小盆友优先程度很高啊)



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

- 把堂兄弟/姐妹 和 Parch 还有自己 个数加在一起组一个Family_size字段(考虑到大家族可能对最后的结果有影响)
- Name是一个我们一直没有触碰的属性，我们可以做一些简单的处理，比如说男性中带某些字眼的(‘Capt’ , ‘Don’ , ‘Major’ , ‘Sir’)可以统

大家接着往下挖掘，可能还可以想到更多可以细挖的部分。我这里先列这些了，然后我们可以使用手头上的“ train_df” 和“ cv_df” tricks是否有效了。

试验的过程比较漫长，也需要有耐心，而且我们经常会面临很尴尬的状况，就是我们灵光一闪，想到一个feature，然后坚信它一定会有之前的结果。恩，需要坚持和耐心，以及不断的挖掘。

我最好的结果是在『Survived~C(Pclass)+C(Title)+C(Sex)+C(Age_bucket)+C(Cabin_num_bucket)Mother+Fare+Family_Size』，在提交的时候手抖把页面关了，于是没截着图，下面这张图是在我得到最高分之后，用这次的结果重新make commission的，截了个最高分哈，因此排名木有变...）：

691	new	Tamas S.	0.80383	4	Tue, 10 Nov 2015 20:45:41 (-0.1h)
692	new	Jack Collins	0.80383	8	Wed, 11 Nov 2015 21:58:38 (-22.7h)
693	11353	BillyFung	0.80383	3	Wed, 11 Nov 2015 00:17:25
694	829	JyothiPriyan	0.80383	4	Wed, 11 Nov 2015 03:10:16
695	new	hanxiaoyang	0.80383	9	Thu, 12 Nov 2015 03:27:44 (-13.9h)
Your Best Entry ↑ Your submission scored 0.79426, which is not an improvement of your best score. Keep trying!					
696	new	dgt01	0.80383	3	Wed, 11 Nov 2015 16:41:58 (-0.9h)
697	new	Roman Minko	0.80383	43	Wed, 11 Nov 2015 16:26:47
698	30	yoTTaBo55	0.80383	16	Wed, 11 Nov 2015 17:37:35
699	11228	rehlezen	0.80383	14	Thu, 12 Nov 2015 01:06:46 (-6.5h)

9.3 learning curves

有一个很可能发生的问题是，我们不断地做feature engineering，产生的特征越来越多，用这些特征去训练模型，会对我们的训练集拟合得越来越好，从而在待预测的数据上，表现不佳，也就是发生过拟合问题。

从另一个角度上说，如果模型在待预测的数据上表现不佳，除掉上面说的过拟合问题，也有可能是欠拟合问题，也就是说在训练集上，其实拟合的还不错，这个欠拟合和过拟合怎么解释呢。这么说吧：

- 过拟合就像是你们班那个学数学比较刻板的同学，老师讲过的题目，一字不漏全记下来了，于是老师再出一样的题目，分分钟精确出结果。但是数学考试，因为总是新题。
- 欠拟合就像是，咳咳，和博主level差不多的差生。连老师讲的练习题也记不住，于是连老师出一样题目复习的周测都做不好，考试更是可想而知了。

而在机器学习的问题上，对于过拟合和欠拟合两种情形。我们优化的方式是不同的。

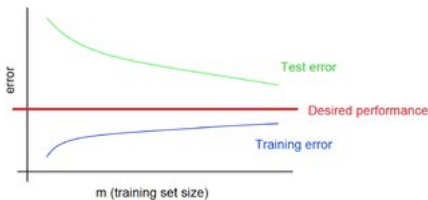
对过拟合而言，通常以下策略对结果优化是有用的：

- 做一下feature selection，挑出较好的feature的subset来做training
- 提供更多的数据，从而弥补原始数据的bias问题，学习到的model也会更准确

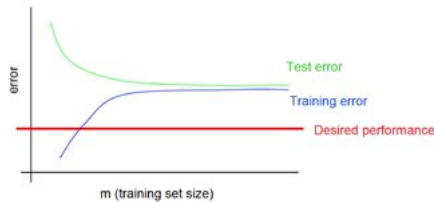
而对于欠拟合而言，我们通常需要更多的feature，更复杂的模型来提高准确度。

著名的learning curve可以帮助我们判定我们的模型现在所处的状态。我们以样本数为横坐标，训练和交叉验证集上的错误率作为纵坐标，两种状态：过拟合(overfitting/high variance)，欠拟合(underfitting/high bias)

Typical learning curve for high variance:



Typical learning curve for high bias:



我们也可以把错误率替换成准确率(得分), 得到另一种形式的learning curve(sklearn 里面是这么做的)。

回到我们的问题, 我们用scikit-learn里面的learning_curve来帮我们分辨我们模型的状态。举个例子, 这里我们一起画一下我们最先urve。

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.learning_curve import learning_curve

# 用sklearn的learning_curve得到training_score和cv_score, 使用matplotlib画出learning curve
def plot_learning_curve(estimator, title, X, y, ylim=None, cv=None, n_jobs=1,
                        train_sizes=np.linspace(.05, 1., 20), verbose=0, plot=True):
    """
    画出data在某模型上的learning curve.
    参数解释
    -----
    estimator : 你用的分类器。
    title : 表格的标题。
    X : 输入的feature, numpy类型
    y : 输入的target vector
    ylim : tuple格式的(ymin, ymax), 设定图像中纵坐标的最低点和最高点
    cv : 做cross-validation的时候, 数据分成的份数, 其中一份作为cv集, 其余n-1份作为training(默认为3份)
    n_jobs : 并行的任务数(默认1)
    """
    train_sizes, train_scores, test_scores = learning_curve(
        estimator, X, y, cv=cv, n_jobs=n_jobs, train_sizes=train_sizes, verbose=verbose)

    train_scores_mean = np.mean(train_scores, axis=1)
    train_scores_std = np.std(train_scores, axis=1)
    test_scores_mean = np.mean(test_scores, axis=1)
    test_scores_std = np.std(test_scores, axis=1)

    if plot:
        plt.figure()
        plt.title(title)
        if ylim is not None:
            plt.ylim(*ylim)
        plt.xlabel(u"训练样本数")
        plt.ylabel(u"得分")
        plt.gca().invert_yaxis()
        plt.grid()

        plt.fill_between(train_sizes, train_scores_mean - train_scores_std, train_scores_mean + train_scores_std,
                        alpha=0.1, color="b")
        plt.fill_between(train_sizes, test_scores_mean - test_scores_std, test_scores_mean + test_scores_std,
                        alpha=0.1, color="r")
        plt.plot(train_sizes, train_scores_mean, 'o-', color="b", label=u"训练集上得分")
        plt.plot(train_sizes, test_scores_mean, 'o-', color="r", label=u"交叉验证集上得分")

        plt.legend(loc="best")

    plt.draw()
    plt.show()
    plt.gca().invert_yaxis()

    midpoint = ((train_scores_mean[-1] + train_scores_std[-1]) + (test_scores_mean[-1] - test_scores_std[-1])) / 2
    diff = (train_scores_mean[-1] + train_scores_std[-1]) - (test_scores_mean[-1] - test_scores_std[-1])
    return midpoint, diff
```



人脸识别!

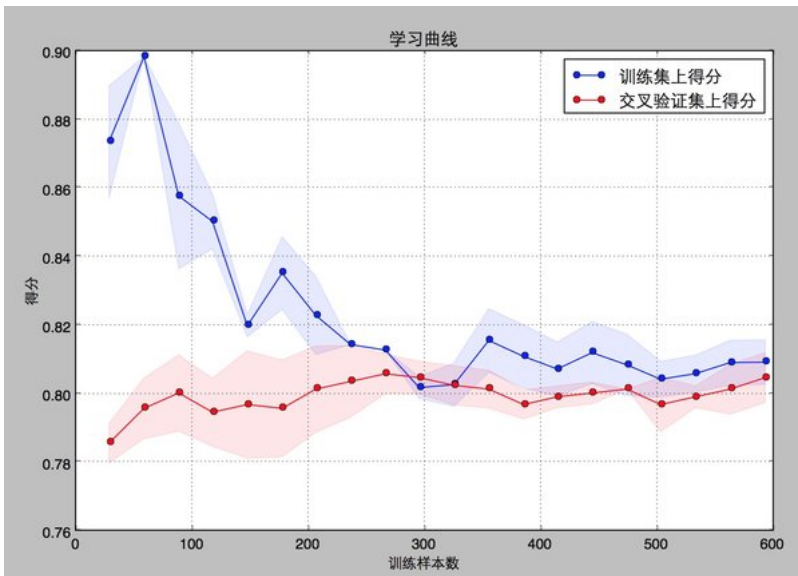


联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网



在实际数据上看，我们得到的learning curve没有理论推导的那么光滑哈，但是可以大致看出来，训练集和交叉验证集上的得分曲线目前的曲线看来，我们的model并不处于overfitting的状态(overfitting的表现一般是训练集上得分高，而交叉验证集上要低很多，中些feature engineering的工作，添加一些新产生的特征或者组合特征到模型中。

10.模型融合(model ensemble)

好了，终于到这一步了，我们要祭出机器学习/数据挖掘上通常最后会用到的大杀器了。恩，模型融合。

『强迫症患者』打算继续喊喊口号...

『模型融合(model ensemble)很重要！』

『模型融合(model ensemble)很重要！』

『模型融合(model ensemble)很重要！』

重要的事情说三遍，恩，噢啦。

先解释解释，一会儿再回到我们的问题上哈。

啥叫模型融合呢，我们还是举几个例子直观理解一下好了。

大家都看过知识问答的综艺节目中，求助现场观众时候，让观众投票，最高的答案作为自己的答案的形式吧，每个人都有一个判定结果，最后我们里。

再通俗一点举个例子。你和你班某数学大神关系好，每次作业都『模仿』他的，于是绝大多数情况下，他做对了，你也对了。突然某一天大神脑子了一个数，于是...恩，你也只能跟着错了。

我们再来看看另外一个场景，你和你班5个数学大神关系都很好，每次都把他们作业拿过来，对比一下，再『自己做』，那你想想，如果哪天某大神另外四个写对了啊，那你肯定相信另外4人的是正确答案吧？

最简单的模型融合大概就是这么个意思，比如分类问题，当我们手头上有一堆在同一份数据集上训练得到的分类器(比如logistic regression, SVM, 神经网络)，那我们让他们都分别去做判定，然后对结果做投票统计，取票数最多的结果为最后结果。

bingo，问题就这么完美的解决了。

模型融合可以比较好地缓解，训练过程中产生的过拟合问题，从而对于结果的准确度提升有一定的帮助。

话说回来，回到我们现在的问题。你看，我们现在只讲了logistic regression，如果我们还想用这个融合思想去提高我们的结果，我们该怎么做呢？

既然这个时候模型没得选，那咱们就在数据上动动手脚咯。大家想想，如果模型出现过拟合现在，一定是在我们的训练上出现拟合过度造成的对吧。

那我们干脆就不要用全部的训练集，每次取训练集的一个subset，做训练，这样，我们虽然用的是同一个机器学习算法，但是得到的模型却是不一没有任一份子数据集是全的，因此即使出现过拟合，也是在子训练集上出现过拟合，而不是全体数据上，这样做一个融合，可能对最后的结果有是常用的Bagging。

我们用scikit-learn里面的Bagging来完成上面的思路，过程非常简单。代码如下：

```
from sklearn.ensemble import BaggingRegressor

train_df = df.filter(regex='Survived|Age_.*|SibSp|Parch|Fare_.*|Cabin_.*|Embarked_.*|Sex_.*|Pclass.*|Mother|Child|Family|Title
```



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

```
y = train_np[:, 0]

# X即特征属性值
X = train_np[:, 1:]

# fit到BaggingRegressor之中
clf = linear_model.LogisticRegression(C=1.0, penalty='l1', tol=1e-6)
bagging_clf = BaggingRegressor(clf, n_estimators=20, max_samples=0.8, max_features=1.0, bootstrap=True, bootstrap_
bagging_clf.fit(X, y)

test = df_test.filter(regex='Age_.*|SibSp|Parch|Fare_.*|Cabin_.*|Embarked_.*|Sex_.*|Pclass.*|Mother|Child|Family|T
predictions = bagging_clf.predict(test)
result = pd.DataFrame({'PassengerId':data_test['PassengerId'].as_matrix(), 'Survived':predictions.astype(np.int32)
result.to_csv("/Users/HanXiaoyang/Titanic_data/logistic_regression_bagging_predictions.csv", index=False)
```

然后你再Make a submission，恩，发现对结果还是有帮助的。

690	↑1466	JamesSeymour	0.80383	82	Wed, 11 Nov 2015 11:33:46 (-17.5h)
691	new	Tamas S.	0.80383	4	Tue, 10 Nov 2015 20:45:41 (-0.1h)
692	new	Jack Collins	0.80383	4	Tue, 10 Nov 2015 23:32:07 (-0.2h)
693	↑1341	BillyFung	0.80383	3	Wed, 11 Nov 2015 00:17:25
694	↑825	JyothiPriyan	0.80383	4	Wed, 11 Nov 2015 03:10:16
695	new	hanxiaoyang	0.80380	6	Wed, 11 Nov 2015 13:35:38
Your Best Entry ↑ You improved on your best score by 0.03823. You just moved up 2,029 positions on the leaderboard. Tweet this!					
696	↓77	tom hacker	0.79904	2	Sat, 12 Sep 2015 17:07:20
697	↓77	mamitasu	0.79904	2	Sat, 12 Sep 2015 17:16:54
698	↓77	zduey	0.79904	6	Sat, 12 Sep 2015 17:42:46
699	↓77	Aishi Jiang	0.79904	10	Sun, 13 Sep 2015 01:09:09 (-0.3h)
700	↓77	Jon Marlow	0.79904	12	Thu, 17 Sep 2015 14:23:48 (-3.6d)

11.总结

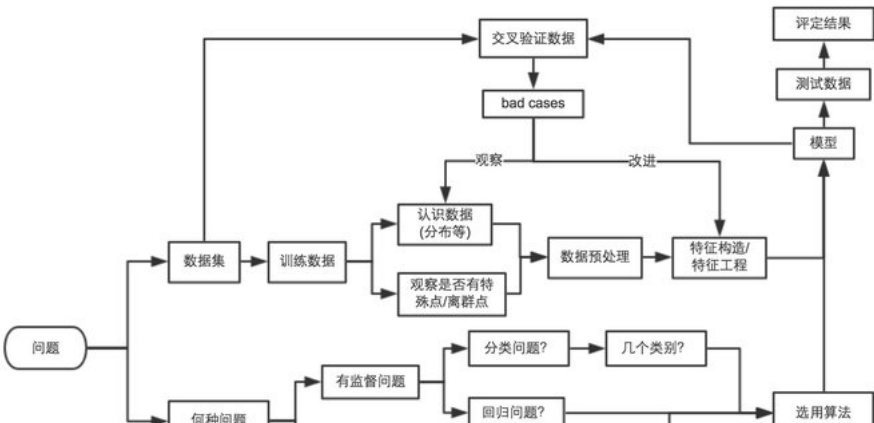
文章稍微有点长，非常感谢各位耐心看到这里。
总结的部分，我就简短写几段，出现的话，很多在文中有对应的场景，大家有兴趣再回头看看。

对于任何的机器学习问题，不要一上来就追求尽善尽美，先用自己会的算法撸一个baseline的model出来，再进行后续的分析步骤，一步步提高。

在问题的结果过程中：

- 『对数据的认识太重要了！』
- 『数据中的特殊点/离群点的分析和处理太重要了！』
- 『特征工程(feature engineering)太重要了！』
- 『模型融合(model ensemble)太重要了！』

本文中用机器学习解决问题的过程大概如下图所示：






联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

想对作者说点什么？

 **Evil_Teddy** 2018-04-18 11:41:29 #4楼
借鉴了，谢谢博主

 **kunduan5343** 2018-04-11 15:25:50 #3楼
博主好强！！学习了

 **shihunyewu** 2018-03-06 15:35:47 #2楼
感谢博主！！！！

查看 5 条热评


Kaggle入门系列：（三）Titanic竞赛初试身手

本篇文章针对刚刚接触Kaggle的同学，介绍参加Kaggle数据分析竞赛的完整流程，最快速度完成一个比赛。...

 **u013115001** 2017-09-17 19:28:20 阅读数：798

Kaggle竞赛——泰坦尼克号（Titanic）

kaggle Titanic数据分析

 **wydyttxs** 2017-08-04 21:03:33 阅读数：2315

Titanic 生存预测详细笔记 - CSDN博客

如何预测Titanic生存率的流程 简介 这是kaggle上入门推荐的测试例子,我将经历在著名的泰坦尼克号数据集上创建一个机器学习模型的全过程,这个模型已经被全世界的机器学习社区广泛使用
2018-6-3

Kaggle_Titanic生存预测 -- 详细流程吐血梳理 - CSDN博客

Kaggle 泰坦尼克生存预测,详细流程分析整理:特征分析,特征工程,数据处理,Stacking框架,xgboost融合
2018-5-30

可惜不在深圳，科技园可免费报读短学制成人本科，快速拿一年名校本科

青学教育 · 顶新

使用sklearn进行kaggle案例泰坦尼克Titanic船员获救预测

数据集下载地址https://github.com/fayduan/Kaggle_Titanic/blob/master/train.csv python代码: #-*- coding: UTF-8 ...

 **qifuchenluo** 2017-03-28 21:59:39 阅读数：7098

kaggle 泰坦尼克号生存预测——六种算法模型实现与比较 - CSDN博客

便参加了kaggle 上最为经典的泰坦尼克号生存预测项目,期间自己探索了一些,也拜读了一些大神的作品,感觉收获蛮大。为了不至于忘掉,在此做个总结,并希望大家交流
2018-6-6

R语言学习:泰坦尼克号生存预测 - CSDN博客

机器学习系列(3)_逻辑回归应用之Kaggle泰坦尼克之灾

作者：寒小阳 &&龙心尘 时间：2015年10月。 出处： 声明：版权所有，转载请注明出处，谢谢。手把手机器学习之逻辑回归应用——Kaggle泰坦尼克之灾

yaoqiang2011 2015-11-12 12:07:12 阅读数：141144

Kaggle: Titanic问题

Kaggle: Titanic问题 相关库函数操作 1.pandas的read_csv函数 读取csv文件为DataFrame格式 from pandas import Data...

qq_37423198 2017-08-06 23:30:33 阅读数：318

泰坦尼克号预测生存概率数据集

举报人: 被举报人: crazybboy 举报的资源分: 3 *类型: *详细原因: 取 消 提 交 泰坦尼克号预测生存概率数据集 3积分 立即下载 ...

2018-5-7

Kaggle: 泰坦尼克号生存预测 - CSDN博客

本文对Kaggle泰坦尼克比赛的训练集和测试集进行分析,并对乘客的生存结果进行了预测.作为数据挖掘的入门项目,本人将思路记录下来,以供参考.如有不

2018-5-11

Kaggle入门——Titanic案例

Kaggle Titanic

Shaaron_Chan 2017-01-12 22:01:33 阅读数：2053

可惜不在深圳，科技园现可免费报读短学制成人本科，今年报名明年取证！

青学教育 · 顶新

sklearn集合算法预测泰坦尼克号幸存者 - CSDN博客

原文: http://ihoge.cn/2018/sklearn-ensemble.html 随机森林分类预测泰坦尼克号幸存者 import pandas as pd import numpy as np def read_dataset(fname): ...

2018-5-23

Python机器学习实践例子——Titanic乘客生存预测模型分..._CSDN博客

是否幸存通过0,1记录,因为sum即为生存的人数,len即为总人数,mean即为生存百分比...最近在学习机器学习,有幸可以研究一下Kaggle竞赛的"泰坦尼克号预测生还"的案例

2018-6-6

Kaggle竞赛-Titanic泰坦尼克

在博主的原有基础上修改了部分错误，Jupyter Notebook实现。代码链接：http://downlo

linxid 2018-01-30 10:07:17 阅读数：191

Kaggle_Titanic生存预测 -- 详细流程吐血梳理

Kaggle 泰坦尼克生存预测，详细流程分析整理：特征分析，特征工程，数据处理，Stacking框架，xgboost融合...

Koala_Tree 2017-12-06 10:55:55 阅读数：7807

泰坦尼克号预测生还案例的分析(一) - CSDN博客

最近在学习机器学习,有幸可以研究一下Kaggle竞赛的"泰坦尼克号预测生还"的案例。首先说明这里的代码并非全部为本人所打,只是在从中学学习到了很多东西,一和大家

2018-6-10

kaggle竞赛入门Titanic生存预测 - CSDN博客

Titanic是kaggle上的一道入门题目,很适合新手去练手数据分析。这道题给的数据是泰坦尼克号上的乘客的信息,预测乘客是否幸存。这是个二元分类的机器学习问题。数

2018-5-23



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

 qjc937044867 2016-03-02 08:49:01 阅读数：2117

kaggle titanic 机器学习流程 top30%

机器学习流程解决kaggle上的Titanic 问题。通过数据分析、特征工程、特征选择、模型调优、模型融合等过程。...

 cqy_chen 2017-12-12 16:59:44 阅读数：557


机器学习实战（三）kaggle titanic随机森林

一、问题描述 kaggle平台是一个大数据商业化平台，在平台上有很多lab和企业会提出很多问题，解答相应的问题也许会有相应的奖金。本文中的titanic是分享乘客...

 u014744118 2017-11-12 22:14:55 阅读数：358

Kaggle系列——Titanic 80% + 精确度纪录

因为最近模型上线收益没有符合预期，一直都没有时间搞，而且感觉Titanic的数据量太少了，做起来没意思，暂且优化到0.8的precision，排名700 + ...

 yobobobo 2015-09-04 22:24:40 阅读数：14025

机器学习一小步：Kaggle上的练习Titanic: Machine Learning from Disaster（一）

来自Kaggle上的练习Titanic: Machine Learning from Disaster，题目的大意是当年泰坦尼克号的沉没造成了很多人的死亡，其中比较重要的一个因素是救

 SilenceGTX 2016-02-19 14:05:38 阅读数：7234

TEK无线吸尘器618立减300咨询再送券

618狂欢价1 ? 99元>

kaggle titanic数据

2017年07月31日 32KB [下载](#)

kaggle入门竞赛--Titanic：Machine Learning from Disaster

Titanic是Kaggle入门竞赛的第一个问题，泰坦尼克号已经是众所周知的事情，在这场灾难中存活下来的人非常少，运用机器学习的知识预测某个人在这场灾难中是否能活过这道题一共做了三...

 yimi1995 2017-07-14 17:15:49 阅读数：782

用python实现Kaggle的Titanic数据分析例子

一、在数据处理方向上，R语言相比，python更接近编程语言，先学习pandas包的内容，之后再学习sklearn包运用，先这样打算吧！我竟然连一次完整的模型构建都没搞...

 yezonggang 2016-07-04 23:44:00 阅读数：11530

kaggle Titanic泰坦尼克

作者：寒小阳 出处：http://blog.csdn.net/han_xiaoyang/article/details/49797143 1.引言先说一句，年末双十一什么的一来，真是非(man...

 weiyongle1996 2017-09-20 10:58:03 阅读数：1314

kaggle比赛泰坦尼克号基于别人的处理流程的学习总结

本篇是基于kaggle上一位作者的学习流程学习所写的总结，应用的都是一些比较基础的操作，主要发力在数据处理方面，对算法的优化过程几乎没有，下面是原文链接...

 jibiqu2893 2018-04-19 13:32:08 阅读数：78

什么叫工业设计

工业设计

百度广告



kaggle——泰坦尼克号生死预测



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

Kaggle比赛经验总结之Titanic: Machine Learning from Disaster

这是个根据旅客信息判断他是否幸存的二分类问题，适合机器学习新手入门。本文记录了我大致的处理思路和步骤，然后总结了一下经验，同时有助于特80383 代码...

 login_sonata

2017-01-10 16:39:49

阅读数：2390

kaggle泰坦尼克数据titanic

2017年08月16日

32KB

下载

数据分析kaggle大赛泰坦尼克号数据

2017年07月28日

32KB

下载

kaggle数据挖掘竞赛初步--Titanic<数据变换> , kaggle--titanic

kaggle数据挖掘竞赛初步--Titanic , kaggle--titanic 完整代码： <https://github.com/cindycindyhi/kaggle-Titanic> ...

 u011089523

2017-05-17 13:47:21

阅读数：662

TEK无线吸尘器618立减700 买贵退差

618狂欢价1 ? 99元>

Titanic: Machine Learning from Disaster (Kaggle 数据挖掘竞赛)

Predict survival on the Titanic (with tutorials in Excel, Python, R, and an introduction to Random F...

 Wiking__acm

2015-01-15 16:27:10

阅读数：5680

Kaggle竞赛：泰坦尼克号灾难数据分析简单案例

Kaggle竞赛：泰坦尼克号灾难数据分析<https://www.kaggle.com/c/titanic> 目标确定：根据已有数据预测未知旅客生死 数据准备：数据获取，载入训练集csv、测试集csv.


 QianYanDai

2017-09-12 21:35:02

阅读数：1405

Kaggle/Titanic python分析和建模

Titanic是Kaggle入门项目，本文跟随<https://www.kaggle.com/startupsci/titanic/titanic-data-science-solutions>学习。 ...

 sunfoot001

2017-07-29 21:28:21

阅读数：410

Kaggle实例-Titanic分析（一）

看了很多发掘类和ML的文章，但是一直苦于没有想法，没有进行过想关的尝试以及练习，无意中在csdn中看到了hanxiaoyang大佬的一篇初学者测试的文章，对照着dal的首次尝试...

 Exziro

2017-09-14 22:42:07

阅读数：365

kaggle数据挖掘——以Titanic为例介绍处理数据大致步骤

Titanic是kaggle上的一道just for fun的题，没有奖金，但是数据整洁，拿来练手最好不过。本文以 Titanic 的数据，使用较为简单的决策树，介绍处理数据大致过程、步骤

 u012675539

2015-07-28 20:06:55

阅读数：4604

大数据分析工具

关于大数据分析的8大工具

百度广告



【Kaggle】Titanic详解

kaggle： <https://www.kaggle.com/c/titanic>这里做一个简单笔记记录提交准确率：0.83代码详解:1、数据读取#读取训练集 train = pd.read_csv(...

 cheneykl

2018-03-22 17:44:21

阅读数：106



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

 helloworld_Fly 2017-11-07 20:34:16 阅读数：251

tensorflow实现逻辑回归，在kaggle《泰坦尼克》训练并测试准确率

tensorflow实现逻辑回归，在kaggle《泰坦尼克》训练并测试准确率 主要有以下3个步骤： 1 数据集特征分析、预处理 2 基于tensorflow的逻辑回归 3 训练

 yimi_ac 2018-01-09 00:18:04 阅读数：884


Kaggle Titanic 机器学习实践笔记（二）

Kaggle titanic 数据预处理 填补空缺值 baseline_model learning_curve

 u011462357 2017-12-04 22:04:42 阅读数：105

Kaggle Titanic 机器学习实践笔记

目录结构是这个样子的： Titanic --data (放官网的数据) --train.csv --test.csv --gender_submission....

 u011462357 2017-12-04 17:02:11 阅读数：239

50万码农评论：英语对于程序员有多重要？

不背单词和语法，一个公式学好英语

kaggle titanic 入门实例 逻辑回归的使用 & 随机森林的使用

```
#coding:utf-8 import numpy as np import pandas as pd train = pd.read_csv("./csv/train.csv", dtype={"A...
```

 guotong1988 2016-01-11 11:58:47 阅读数：2739

kaggle 案例之泰坦尼克号船员获救预测

使用Python数据分析最流行的库Numpy,Pandas,Matplotlib, Scikit-learn结合真实数据集展开可视化特征分析与机器学习建模和评估。每次课程涉及一个完整的案例，基于
学院 2017年02月28日 13:32

kaggle系列（一、Titanic入门比赛）

Table of Contents 1 背景介绍2 数据导入与分析 2.1 导入有用的包2.2 导入数据2.3 去除离群点2.4 连接训练数据和测试数据2.5 查看缺失值 3 ...

 wl_ss 2017-11-15 16:54:45 阅读数：762

kaggle入门泰坦尼克之灾内容总结

[泰坦尼克之灾](https://www.kaggle.com/c/titanic)：求生问题预测，是一个二分类问题，判断每名乘客的存活情况，读了寒小阳大牛的博客，现做以下整理。阳哥只介绍

 gufeng_1992 2017-05-30 15:14:27 阅读数：1372

Kaggle中Titanic项目简单入门

首先，再kaggle注册帐号，找到Titanic项目，9659个队伍，估计全部都是菜鸟： 下载train和test集，提交的文件： 首先，二话不说，先把下载的《gender_submiss...

 masbbx123 2018-02-11 17:06:19 阅读数：167

资产管理系统

资产管理系统解决方案

百度广告



Kaggle入门之泰坦尼克号生还率预测

这是Kaggle上的一道入门题目，旨在让我们了解机器学习的大致过程。 题目链接：Titanic: Machine Learning from Disaster 题目大意：当年泰坦尼克号的沉没造成了很

 qq_26658823 2017-12-17 16:33:20 阅读数：564

Kaggle-Titanic一个完整的例子

看了Kaggle的Titanic练手项目kernel中的4个教程，自己写了一个总结

 Asun0204 2017-09-15 13:05:01 阅读数：588




联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

Kaggle数据竞赛是学习ML的一种有效途径。

 u011933487 2014-09-01 16:59:48 阅读数：22596

kaggle 泰坦尼克号生存预测——六种算法模型实现与比较

Hi，大家好，这是我第一篇博客。作为非专业程序小白，博客内容必然有不少错误之处，还望各位大神多多批评指正。在开始正式内容想先介绍下自己和生，研究的方向是蛋白...

 aicanghai_smile 2018-02-04 15:56:23 阅读数：796

数据挖掘实战系列 之 Kaggle 泰坦尼克号灾难（上）

（一）步骤流程：#（一）目标确定：根据已有数据预测未知旅客生死#（二）数据准备：1 数据获取，载入训练集csv，测试集csv#（三）数据清洗：串...

 xudailong_blog 2018-05-07 23:37:20 阅读数：55

50万码农评论：英语对于程序员有多重要！

不背单词和语法，老司机教你一个数学公式秒懂天下英语

机器学习基础—Kaggle泰坦尼克预测(完整分析)

1.引言我们先找个简单的实际例子，来看看，所谓的数据挖掘或者机器学习实际应用到底是怎么样一个过程。2.背景2.1 关于KaggleKaggle是一个数据分UP（...

 xun527 2017-11-20 00:01:16 阅读数：277

Kaggle-Titanic入门教程1

序言 这篇文章翻译自Titanic Data Science Solutions 这篇文章带我们浏览了对于解决Kaggle这样的数据科学竞赛的典型工作流程。在Kaggle的Kernels里有很...

 gemmax 2018-05-28 23:14:05 阅读数：13

Kaggle入门项目，泰坦尼克号幸存者

泰坦尼克号幸存者项目是kaggle的入门项目，我先用python的matplotlib库对数据进行了可视化，初步探索后对数据进行了清洗，然后建立了逻辑回归模型对测试集

 weixin_42036641 2018-05-14 10:42:20 阅读数：79

kaggle竞赛题 泰坦尼克号生存者预测

由于各种不可抗力，笔者在学习完numpy，pandas，和一些sklearn知识后，不得不开始泰坦尼克号生存者预测。但由于经验不足，掌握的知识不足，所以跟着知乎上一教学一步一步的...

 s1347563786 2018-05-11 15:50:56 阅读数：38

kaggle 泰坦尼克号数据分析 笔记

#泰坦尼克号 <https://www.kaggle.com/> 数据来源 import pandas #读取数据 titanic = pandas.read_csv(""D:/panana/ta...

 shiershier123 2018-03-15 22:48:15 阅读数：64

什么叫工业设计

工业设计

百度广告



泰坦尼克号预测生还案例的分析（一）

最近在学习机器学习，有幸可以研究一下Kaggle竞赛的“泰坦尼克号预测生还”的案例。首先说明这里的代码并非全部为本人所打，只是在从中学学习到了很多东西，一和识。先分享下的数据文件：...

 ZengHaihong 2016-11-17 14:30:44 阅读数：5417

【机器学习实战】决策树预测Titanic遇难者生还情况

一、导入数据#导入pandas用于数据分析 import pandas as pd #利用pandas的read_csv模块直接从互联网手机泰坦尼克号乘客数据 titanic = pd.read_cs...

 chihangvuxin 2017-03-05 11:40:38 阅读数：3123



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站
网络110报警
中国互联网
北京互联网

个人资料



珠穆拉玛峰

原创17

粉丝36

喜欢27

等级：博客4

访问：10万+

积分：1348

排名：3万+



在职研究生取消



最新文章

隐因子分解机 (Factorization Machine)

python画图常用颜色

从拉普拉斯矩阵说到谱聚类

模型融合

xgboost公式推导

个人分类

ASP.NET学习1篇

hadoop学习4篇

软件安装配置20篇

设计模式2篇

模式识别1篇

展开

归档

2018年5月2篇

2018年3月1篇

2018年2月2篇

2017年12月4篇

2017年11月2篇

展开

热门文章

PCA(协方差矩阵和奇异值分解两种方法求特征值特征向量)
阅读量：9157

Kaggle泰坦尼克预测(完整分析)
阅读量：8043



人脸识别!



联系我们



关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网

阅读量：7094

推荐系统中SVD算法详解

阅读量：6560

sed 正则替换

阅读量：6187



最新评论

Ubuntu修改桌面大小
masuwen：好好好好,所得太好了,方案三
能用!!!!

pandas基本命令操作
renjingyuan：head()我记忆中无参数是显
行？

Kaggle泰坦尼克预测(完整分析)
weixin_40900175：借鉴了，谢谢博主

Kaggle泰坦尼克预测(完整分析)
kunduan5343：博主好强！！学习了

Kaggle泰坦尼克预测(完整分析)
weixin_40570339：[reply]Bobby_world[/i
小白，盖楼，留名~~

联系我们

关于 招聘
©2018 CSDN
百度提供

经营性网站备案
网络110报警
中国互联网
北京互联网