Pandas 实战 Kaggle titanic 幸存预测之 7 步数据清洗

数据清洗 (data cleaning) 是机器学习和深度学习进入算法步前的一项重要任务，总结为下面几个步骤。

步骤 1：读入 csv 数据；

步骤 2：预览数据；

步骤 3：统计每一列的空值；

步骤 4：填充空值

步骤 5：特征工程，子步骤包括：删除一些特征列；创建新的特征列；创建数据分箱；

步骤 6：对分类列编码，常用的包括，调用 Sklearn 中 LabelEncode 编码 ；Pandas 中哑编码；

步骤 7：再验证核实

今天使用泰坦尼克数据集，完整介绍以上步骤的具体操作过程。

1 读入数据

使用 Pandas, 读入 csv 训练数据，然后了解每个字段的含义，数据有多少行和多少列等。

复制

```
import pandas as pd

data_raw = pd.read_csv('train.csv')
data_raw
```

结果如下，一共训练集有 891 行数据，12 列

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows × 12 columns

PassengerId: 乘客的 Id；

Survived：乘客生还情况，取值 1,2；

Pclass：乘客等级，取值：1,2,3；

SibSp：乘客的兄弟姐妹和配偶在船上的人数；

Parch：乘客的父母和孩子在船上的人数；

Fare：乘船的费用；

Cabin：舱的编号；

Embarked：分类变量，取值 S, C, Q；

其他几个特征比较好辨别，不再解释。

2 数据预览

Pandas 提供 2 个好用的方法：`info`，`describe`

`info` 统计出数据的每一列类型、是否为 null 和个数；

`describe` 统计出数据每一列的统计学属性信息，平均值，方差，中位数，分位数等。

复制

```
data_raw.info()
data_raw.describe(include='all')
```

结果：

复制

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
PassengerId    891 non-null int64
Survived       891 non-null int64
Pclass         891 non-null int64
Name           891 non-null object
Sex            891 non-null object
Age            714 non-null float64
SibSp          891 non-null int64
Parch          891 non-null int64
Ticket         891 non-null object
Fare           891 non-null float64
Cabin          204 non-null object
Embarked       889 non-null object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
count	891.000000	891.000000	891.000000	891	891	714.000000	891.000000	891.000000	891	891.000000	204	889
unique	NaN	NaN	NaN	891	2	NaN	NaN	NaN	681	NaN	147	3
top	NaN	NaN	NaN	Richards, Master. William Rowe	male	NaN	NaN	NaN	CA. 2343	NaN	C23 C25 C27	S
freq	NaN	NaN	NaN	1	577	NaN	NaN	NaN	7	NaN	4	644
mean	446.000000	0.383838	2.308642	NaN	NaN	29.699118	0.523008	0.381594	NaN	32.204208	NaN	NaN
std	257.353842	0.486592	0.836071	NaN	NaN	14.526497	1.102743	0.806057	NaN	49.693429	NaN	NaN
min	1.000000	0.000000	1.000000	NaN	NaN	0.420000	0.000000	0.000000	NaN	0.000000	NaN	NaN
25%	223.500000	0.000000	2.000000	NaN	NaN	20.125000	0.000000	0.000000	NaN	7.910400	NaN	NaN
50%	446.000000	0.000000	3.000000	NaN	NaN	28.000000	0.000000	0.000000	NaN	14.454200	NaN	NaN
75%	668.500000	1.000000	3.000000	NaN	NaN	38.000000	1.000000	0.000000	NaN	31.000000	NaN	NaN
max	891.000000	1.000000	3.000000	NaN	NaN	80.000000	8.000000	6.000000	NaN	512.329200	NaN	NaN

3 检查 null 值

实际使用的数据，null 值在所难免。如何快速找出 DataFrame 每一列的 null 值个数？

使用 Pandas 能非常方便实现，只需下面一行代码：

复制
```
data1_null = data1.isnull().sum()
```

data.isnull() : 逐行逐元素查找元素值是否为 null.

sum(): 默认在 axis 为 0 上完成一次 reduce 求和。

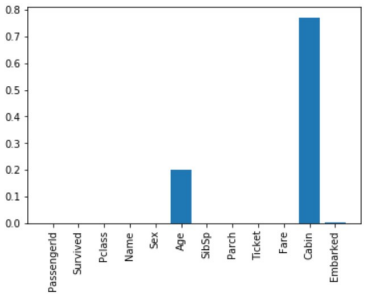
结果：

复制
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

查看每列的空值占比：

复制
```
import matplotlib.pyplot as plt
import seaborn as sns

x_raw = data1.columns
null_rate = data1_null.values / len(data1)
plt.bar(x_raw,null_rate)
plt.xticks(rotation=90)
plt.show()
```



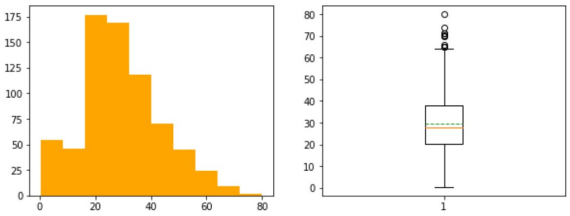
Cabin 列空值比重最大，接近 80%，如此大的空值占比，可以直接删除。

4 补全空值

Age 列和 Embarked 列也存在空值，空值一般用此列的平均值、中位数、众数等填充。

观察 Age 列的取值分布直方图和箱型图

复制
```
plt.figure(figsize=[10,8])
notnull_age_index = data1['Age'].notnull()
plt.subplot(221)
plt.hist(x = data1[notnull_age_index]['Age'], color = ['orange'])
plt.subplot(222)
plt.boxplot(x = data1[notnull_age_index]['Age'], showmeans = True, meanli
ne = True)
plt.show()
```



集中在 20-40 岁，使用中位数填充空值：

复制
```
data1['Age'].fillna(data1['Age'].median(), inplace = True)
```

Embarked 属于分类型变量，使用众数填充：

复制
```
data1['Embarked'].fillna(data1['Embarked'].mode()[0], inplace = True)
```

填充完成后，检查这两列的空值是否全部填充成功。

复制
```
data1.isnull().sum()
```

复制
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         0
dtype: int64
```

5 特征工程

完成数据的基本填充后，下面开始使用 Pandas 做特征工程。

5.1 删除特征列

因为 Cabin 缺失率较大，所以直接删除此列。

使用 Pandas 删除 列 `Cabin`，`axis` 参数设置为 1，表示轴为列方向，`inplace` 为 True 表示就地删除。

复制
```
data1.drop('Cabin', axis=1, inplace = True)
```

另外两列，PassengerId, Ticket 都是 ID 类型的，对预测乘客能否逃离没有关系，也直接删除。

复制
```
drop_column = ['PassengerId','Ticket']
data1.drop(drop_column, axis=1, inplace = True)
```

5.2 增加 3 个特征列

增加一列 `FamilySize` ，计算公式如下：

复制
```
data1['FamilySize'] = data1 ['SibSp'] + data1['Parch'] + 1
data1.head(3)
```

再创建一列 `IsAlone` ，如果 `FamilySize` 为 0，则表示只有一个人，`IsAlone` 为 True.

应用前面介绍的 where 函数，非常简洁地实现 `IsAlone` 列的赋值。

复制
```
data1['IsAlone'] = np.where(data1['FamilySize'] > 1,0,1)
```

再创建一列 `Title`，它是从 Name 列中提取出头衔或称谓。

Name 列的前三行，如下：

复制
```
0                              Braund, Mr. Owen Harris
1    Cumings, Mrs. John Bradley (Florence Briggs Th...
2                               Heikkinen, Miss. Laina
Name: Name, dtype: object
```

Pandas 中使用 `str` 属性，直接拿到整个此列的字符串值，然后使用前面介绍的字符串分隔方法 `split`

复制
```
data1['Title'] = data1['Name'].str.split(", ", expand=True)[1].str.split(
".", expand=True)[0]
data1
```

数据前三行使用上面代码，提取后结果如下：

复制
```
0      Mr
1     Mrs
2    Miss
Name: 0, dtype: object
```

5.3 分箱

Pandas 提供两种数据分箱方法：`qcut`, `cut`.

`qcut` 方法是基于分位数的分箱技术，cut 基于区间长度切分为若干。使用方法如下：

复制
```
a = [3,1,5, 7,6, 5, 4, 6, 3]
pd.qcut(a,3)
```

结果如下，共划分为 3 个分类：

复制
```
[(0.999, 3.667], (0.999, 3.667], (3.667, 5.333], (5.333, 7.0], (5.333, 7.
0], (3.667, 5.333], (3.667, 5.333], (5.333, 7.0], (0.999, 3.667]]
Categories (3, interval[float64]): [(0.999, 3.667] < (3.667, 5.333] < (5.
333, 7.0]]
```

a 元素这 3 个分类中的个数相等：

复制
```
dfa = pd.DataFrame(a)
len1 = dfa[(0.999 < dfa[0]) & (dfa[0] <= 3.667)].shape
len2 = dfa[(3.667 < dfa[0]) & (dfa[0] <= 5.333)].shape
len3 = dfa[(5.333 < dfa[0]) & (dfa[0] <= 7.0)].shape
len1,len2,len3
```

结果如下，每个区间内都有 3 个元素

复制
```
((3, 1), (3, 1), (3, 1))
```

`cut` 方法：

复制
```
a = [3,1,5, 7,6, 5, 4, 6, 3]
pd.cut(a,3)
```

得到结果，与 `qcut` 划分出的 3 个区间不同，cut 根据 a 列表中最大与最小值间隔，均分，第一个左区间做一定偏移。

复制
```
[(0.994, 3.0], (0.994, 3.0], (3.0, 5.0], (5.0, 7.0], (5.0, 7.0], (3.0, 5.
0], (3.0, 5.0], (5.0, 7.0], (0.994, 3.0]]
Categories (3, interval[float64]): [(0.994, 3.0] < (3.0, 5.0] < (5.0, 7.0
]]
```

除此之外，1992 年 Kerber 在论文中提出 ChiMerge 算法，自底向上的先分割再合并的分箱思想，具体算法步骤：

1) 设置 step 初始值

2) while 相邻区间的 merge 操作：

- 计算相邻区间的卡方值
- 合并卡方值最小的相邻区间
- 判断：是否所有相邻区间的卡方值都大于阈值，若是 break，否则继续 merge.

论文中 m 取值为 2 ，即计算 2 个相邻区间的卡方值，计算方法如下：

k : 类别个数

Ri : 第 i 个分箱内样本总数

Cj : 第 j 类别的样本总数

$$\chi^2 = \sum_{i=1}^{2}\sum_{j=1}^{k}\frac{(A_{ij}-E_{ij})^2}{E_{ij}}$$

$$E_{ij} = R_i * C_j / N$$

$$R_i = \sum_{j=1}^{k} A_{ij}（在i区间的j类别数）$$

$$C_j = \sum_{i=1}^{2} A_{ij}（j类别样本个数）$$

$$N = \sum_{i=1}^{2} R_i（总样本数）$$

分别对 Fare 和 Age 列使用 `qcut`, `cut` 完成分箱，分箱数分别为 4 份，6 份。

复制

```
data1['FareCut'] = pd.qcut(data1['Fare'], 4)
data1['AgeCut'] = pd.cut(data1['Age'].astype(int), 6)
data1.head(3)
```

结果：

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Fare	Embarked	FamilySize	IsAlone	Title	FareCut	AgeCut
0	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	7.2500	S	2	0	Mr	(-0.001, 7.91]	(13.333, 26.667]
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	71.2833	C	2	0	Mrs	(31.0, 512.329]	(26.667, 40.0]
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	7.9250	S	1	1	Miss	(7.91, 14.454]	(13.333, 26.667]

6 编码

本节介绍 2 种常用的分类型变量编码方法，一种是分类变量直接编码，LabelEncoder；另一种对分类变量创建哑变量(dummy variables).

6.1 LabelEncoder 方法

使用 Sklearn 的 LabelEncoder 方法，对分类型变量完成编码。

复制

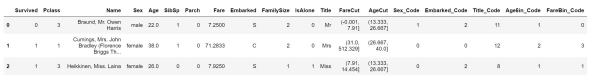
```
from sklearn.preprocessing import LabelEncoder
```

泰坦尼克预测数据集中涉及的分类型变量有： `Sex`, `Embarked`, `Title`，还有我们新创建的2 个分箱列： `AgeCut`, `FareCut`.

复制

```
label = LabelEncoder()
data1['Sex_Code'] = label.fit_transform(data1['Sex'])
data1['Embarked_Code'] = label.fit_transform(data1['Embarked'])
data1['Title_Code'] = label.fit_transform(data1['Title'])
data1['AgeBin_Code'] = label.fit_transform(data1['AgeCut'])
data1['FareBin_Code'] = label.fit_transform(data1['FareCut'])
data1.head(3)
```

使用 LabelEncoder 完成编码后，数据的前三行打印显示：



6.2 get_dummies 方法

Pandas 的 get_dummies 方法，也能实现对分类型变量实现哑编码， 将长 DataFrame 变为宽 DataFrame.

数据集中 `Sex` 分类型列取值有 2 种： female, male DataFrame:

复制

```
pd.get_dummies(data1['Sex'])
```

使用 get_dummies ，返回 2 列，分别为 female, male 列，结果如下：

复制

```
female    male
0    0   1
1    1   0
2    1   0
3    1   0
4    0   1
...    ... ...
886    0   1
887    1   0
888    1   0
889    0   1
890    0   1
891 rows × 2 columns
```

而 LabelEncoder 编码后，仅仅是把 Female 编码为 0， male 编码为 1.

复制

```
label.fit_transform(data1['Sex'])
```

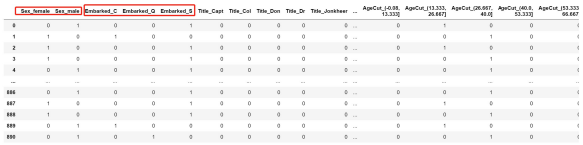
```
0      1
1      0
2      0
3      0
4      1
      ..
886    1
887    0
888    0
889    1
890    1
Name: Sex_Code, Length: 891, dtype: int64
```

对以下变量实现哑编码：

复制

```
data1_dummy = pd.get_dummies(data1[['Sex', 'Embarked', 'Title','AgeCut','
FareCut']])
data1_dummy
```

结果：



	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S	Title_Capt	Title_Col	Title_Don	Title_Dr	Title_Jonkheer	...	AgeCut_(-0.08, 13.333]	AgeCut_(13.333, 26.667]	AgeCut_(26.667, 40.0]	AgeCut_(40.0, 53.333]	AgeCut_(53.333, 66.667]
0	0	1	0	0	1	0	0	0	0	0	...	0	1	0	0	0
1	1	0	1	0	0	0	0	0	0	0	...	0	0	1	0	0
2	1	0	0	0	1	0	0	0	0	0	...	0	1	0	0	0
3	1	0	0	0	1	0	0	0	0	0	...	0	0	1	0	0
4	0	1	0	0	1	0	0	0	0	0	...	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	0	1	0	0	1	0	0	0	0	0	...	0	0	1	0	0
887	1	0	0	0	1	0	0	0	0	0	...	0	1	0	0	0
888	1	0	0	0	1	0	0	0	0	0	...	0	0	1	0	0
889	0	1	1	0	0	0	0	0	0	0	...	0	1	0	0	0
890	0	1	0	1	0	0	0	0	0	0	...	0	0	1	0	0

7 再 次检查

复制

```
data1.info()
print('-'*10)
data1_dummy.info()
```

8 小结

今天与大家一起实践泰坦尼克预测数据集，数据清洗任务。主要使用 Pandas，任务分为 7 个步骤：

- 1) 读入数据
- 2) 数据预览，info, describe 方法
- 3) `isnull()` 检查空值
- 4) `fillna()` 填充空值
- 5) 特征工程，删除和增加特征， 数据分箱：`qcut`，`cut`，`chimerge`算法
- 6) 2 种常见的分类型变量编码方法：`LabelEncoder`，`get_dummies` 方法

下一章

互动评论

说点什么

评论

The Scrapper 1 个月前

跟60天的一样

鼓掌

+ 1 个月前

练习数据有么？

鼓掌

存