利用 Pandas 进行分类数据编码的十种方式!

法纳斯特 2022-04-24 14:00

以下文章来源于早起Python，作者刘早起

早起Python

点击领取pandas数据分析300题

大家好，我是小F。

最近在知乎上看到这样一个问题

题主表示 `pandas` 用起来很乱，事实真的如此吗？本文就将先如何利用 `pandas` 来行数据转换/编码的十种方案，最后再回答这个问题。

其实这个操作在机器学习中十分常见，很多算法都需要我们对分类特征进行转换（编码），即根据某一列的值，新增（修改）一列。

为了方便理解，下面创建示例 `DataFrame`

	Sex	Course Name	Score
0	Male	Python	95
1	Female	Java	85
2	Male	C	75
3	Male	Sql	65
4	Male	Linux	55
5	Female	Python	95
6	Male	Python	75
7	Male	Java	65
8	Female	C	55
9	Female	Php	85

数值型数据

让我们先来讨论连续型数据的转换，也就是根据 `Score` 列的值，来新增一列标签，即如果分数大于90，则标记为A，分数在80-90标记为B，以此类推。

自定义函数 + 循环遍历

首先当然是最简单，最笨的方法，自己写一个函数，并用循环遍历，那肯定就是一个 `def` 加一个 `for`

```
df1 = df.copy()

def myfun(x):
    if x>90:
        return 'A'
    elif x>=80 and x<90:
        return 'B'
    elif x>=70 and x<80:
        return 'C'
    elif x>=60 and x<70:
```

```
        return 'D'
    else:
        return 'E'

df1['Score_Label'] = None
for i in range(len(df1)):
    df1.iloc[i,3] = myfun(df1.iloc[i,2])
```

这段代码，相信所有人都能看懂，简单好想但比较麻烦

	Sex	Course Name	Score	Score_Label
0	Male	Python	95	A
1	Female	Java	85	B
2	Male	C	75	C
3	Male	Sql	65	D
4	Male	Linux	55	E
5	Female	Python	95	A
6	Male	Python	75	C
7	Male	Java	65	D
8	Female	C	55	E
9	Female	Php	85	B

有没有更简单的办法呢？ pandas 当然提供了很多高效的操作的函数，继续往下看。

自定义函数 + map

现在，可以使用 map 来干掉循环（虽然本质上也是循环）

```
df2 = df.copy()

def mapfun(x):
    if x>90:
        return 'A'
    elif x>=80 and x<90:
        return 'B'
    elif x>=70 and x<80:
        return 'C'
    elif x>=60 and x<70:
        return 'D'
    else:
        return 'E'

df2['Score_Label'] = df2['Score'].map(mapfun)
```

结果是同样的

	Sex	Course Name	Score	Score_Label
0	Male	Python	95	A
1	Female	Java	85	B
2	Male	C	75	C
3	Male	Sql	65	D
4	Male	Linux	55	E
5	Female	Python	95	A
6	Male	Python	75	C
7	Male	Java	65	D
8	Female	C	55	E
9	Female	Php	85	B

自定义函数 + apply

如果还想简洁代码，可以使用**自定义函数 + apply**来干掉自定义函数

```
df3 = df.copy()
```

```
df3['Score_Label'] = df3['Score'].apply(lambda x: 'A' if x > 90 else (
    'B' if 90 > x >= 80 else ('C' if 80 > x >= 70 else ('D' if 70 > x >= 60 else 'E'))))
```

结果和上面是一致的，只不过这么写容易被打。

使用 pd.cut

现在，让我们继续了解更高级的 pandas 函数，依旧是对 Score 进行编码，使用 pd.cut ，并指定划分的区间后，可以直接帮你分好组

```
df4 = df.copy()
bins = [0, 59, 70, 80, 100]
df4['Score_Label'] = pd.cut(df4['Score'], bins)
```

	Sex	Course Name	Score	Score_Label
0	Male	Python	95	(80, 100]
1	Female	Java	85	(80, 100]
2	Male	C	75	(70, 80]
3	Male	Sql	65	(59, 70]
4	Male	Linux	55	(0, 59]
5	Female	Python	95	(80, 100]
6	Male	Python	75	(70, 80]
7	Male	Java	65	(59, 70]
8	Female	C	55	(0, 59]
9	Female	Php	85	(80, 100]

也可以直接使用 labels 参数来修改对应组的名称，是不是方便多了

```
df4['Score_Label_new'] = pd.cut(df4['Score'], bins, labels=[
                            'low', 'middle', 'good', 'perfect'])
```

	Sex	Course Name	Score	Score_Label	Score_Label_new
0	Male	Python	95	(80, 100]	perfect
1	Female	Java	85	(80, 100]	perfect
2	Male	C	75	(70, 80]	good
3	Male	Sql	65	(59, 70]	middle
4	Male	Linux	55	(0, 59]	low
5	Female	Python	95	(80, 100]	perfect
6	Male	Python	75	(70, 80]	good
7	Male	Java	65	(59, 70]	middle
8	Female	C	55	(0, 59]	low
9	Female	Php	85	(80, 100]	perfect

使用 sklearn 二值化

既然是和机器学习相关， sklearn 肯定跑不掉，如果需要新增一列并判定成绩是否及格，就可以使用 Binarizer 函数，代码也是简洁好懂

```
df5 = df.copy()
binerize = Binarizer(threshold = 60)
trans = binerize.fit_transform(np.array(df1['Score']).reshape(-1,1))
df5['Score_Label'] = trans
```

	Sex	Course Name	Score	Score_Label
0	Male	Python	95	1
1	Female	Java	85	1
2	Male	C	75	1
3	Male	Sql	65	1
4	Male	Linux	55	0
5	Female	Python	95	1
6	Male	Python	75	1
7	Male	Java	65	1
8	Female	C	55	0
9	Female	Php	85	1

文本型数据

下面介绍更常见的，对文本数据进行转换打标签。例如新增一列，将性别男、女分别标记为0、1

使用 replace

首先介绍 `replace`，但要注意的是，上面说过的自定义函数相关方法依旧是可行的

```
df6 = df.copy()
df6['Sex_Label'] = df6['Sex'].replace(['Male','Female'],[0,1])
```

	Sex	Course Name	Score	Sex_Label
0	Male	Python	95	0
1	Female	Java	85	1
2	Male	C	75	0
3	Male	Sql	65	0
4	Male	Linux	55	0
5	Female	Python	95	1
6	Male	Python	75	0
7	Male	Java	65	0
8	Female	C	55	1
9	Female	Php	85	1

上面是对性别操作，因为只有男女，所以可以手动指定0、1，但要是类别很多，也可以使用 `pd.value_counts()` 来自动指定标签，例如对 `Course Name` 列分组

```
df6 = df.copy()
value = df6['Course Name'].value_counts()
value_map = dict((v, i) for i,v in enumerate(value.index))
df6['Course Name_Label'] = df6.replace({'Course Name':value_map})['Course Name']
```

	Sex	Course Name	Score	Course Name_Label
0	Male	Python	95	0
1	Female	Java	85	1
2	Male	C	75	2
3	Male	Sql	65	5
4	Male	Linux	55	3
5	Female	Python	95	0
6	Male	Python	75	0
7	Male	Java	65	1
8	Female	C	55	2
9	Female	Php	85	4

使用map

额外强调的是，新增一列，一定要能够想到 `map`

```
df7 = df.copy()
Map = {elem:index for index,elem in enumerate(set(df["Course Name"]))}
df7['Course Name_Label'] = df7['Course Name'].map(Map)
```

	Sex	Course Name	Score	Course Name_Label
0	Male	Python	95	5
1	Female	Java	85	2
2	Male	C	75	0
3	Male	Sql	65	3
4	Male	Linux	55	1
5	Female	Python	95	5
6	Male	Python	75	5
7	Male	Java	65	2
8	Female	C	55	0
9	Female	Php	85	4

使用astype

这个方法应该很多人不知道，这就属于上面提到的知乎问题，能实现的方法太多了

```
df8 = df.copy()
value = df8['Course Name'].astype('category')
df8['Course Name_Label'] = value.cat.codes
```

	Sex	Course Name	Score	Course Name_Label
0	Male	Python	95	4
1	Female	Java	85	1
2	Male	C	75	0
3	Male	Sql	65	5
4	Male	Linux	55	2
5	Female	Python	95	4
6	Male	Python	75	4
7	Male	Java	65	1
8	Female	C	55	0
9	Female	Php	85	3

使用 sklearn

同数值型一样，这种机器学习中的经典操作， `sklearn` 一定有办法，使用 `LabelEncoder` 可以对分类数据进行编码

```
from sklearn.preprocessing import LabelEncoder
df9 = df.copy()
le = LabelEncoder()
le.fit(df9['Sex'])
df9['Sex_Label'] = le.transform(df9['Sex'])
le.fit(df9['Course Name'])
df9['Course Name_Label'] = le.transform(df9['Course Name'])
```

	Sex	Course Name	Score	Sex_Label	Course Name_Label
0	Male	Python	95	1	4
1	Female	Java	85	0	1
2	Male	C	75	1	0
3	Male	Sql	65	1	5
4	Male	Linux	55	1	2
5	Female	Python	95	0	4
6	Male	Python	75	1	4
7	Male	Java	65	1	1
8	Female	C	55	0	0
9	Female	Php	85	0	3

一次性转换两列也是可以的

```
df9 = df.copy()
le = OrdinalEncoder()
le.fit(df9[['Sex','Course Name']])
df9[['Sex_Label','Course Name_Label']] = le.transform(df9[['Sex','Course Name']])
```

使用factorize

最后，再介绍一个小众但好用的 `pandas` 方法，我们需要注意到，在上面的方法中，自动生成的 `Course Name_Label` 列，虽然一个数据对应一个语言，因为避免写自定义函数或者字典，这样可以自动生成，所以大多是无序的。

如果我们希望它是有序的，也就是 `Python` 对应 `0` ， `Java` 对应 `1` ，除了自己指定，还有什么优雅的办法？这时可以使用 `factorize` ，它会根据出现顺序进行编码

```
df10 = df.copy()
df10['Course Name_Label'] = pd.factorize(df10['Course Name'])[0]
```

	Sex	Course Name	Score	Course Name_Label
0	Male	Python	95	0
1	Female	Java	85	1
2	Male	C	75	2
3	Male	Sql	65	3
4	Male	Linux	55	4
5	Female	Python	95	0
6	Male	Python	75	0
7	Male	Java	65	1
8	Female	C	55	2
9	Female	Php	85	5

结合匿名函数，我们可以做到对多列进行有序编码转换

```
df10 = df.copy()
cat_columns = df10.select_dtypes(['object']).columns

df10[['Sex_Label', 'Course Name_Label']] = df10[cat_columns].apply(
    lambda x: pd.factorize(x)[0])
```

	Sex	Course Name	Score	Sex_Label	Course Name_Label
0	Male	Python	95	0	0
1	Female	Java	85	1	1
2	Male	C	75	0	2
3	Male	Sql	65	0	3
4	Male	Linux	55	0	4
5	Female	Python	95	1	0
6	Male	Python	75	0	0
7	Male	Java	65	0	1
8	Female	C	55	1	2
9	Female	Php	85	1	5

总结

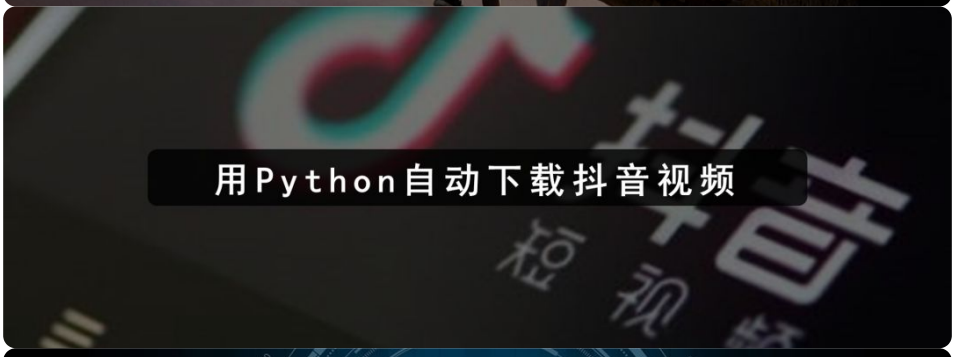
至此，我要介绍的十种 pandas 数据编码的方法就分享完毕，代码拿走修改变量名就能用，关于这个问题如果你有更多的方法，可以在评论区进行留言～

现在回到文章开头的问题，如果你觉得 pandas 用起来很乱，说明你可能还未对 pandas 有一个全面且彻底的了解。

其实就像本文介绍数据编码转换一样，确实有很多方法可以实现显得很乱，但学习 pandas 的正确姿势就是应该把它当成字典来学，不必记住所有方法与细节，**你只需知道有这么个函数能完成这样操作，需要用时能想到，想到再来查就行**。

万水千山总是情，点个 👍 行不行。

推荐阅读



··· END ···

喜欢此内容的人还喜欢

最全面的Python重点知识总结，建议收藏!

法纳斯特