数据预处理 — 编码实践课程

Shùjù yùchǔlǐ — Biānmǎ shíjiàn kèchéng

Data Preprocessing — Encoding Practice Lab

第6周 实践课程 | Python + PyCharm IDE

教师与学生共同体验编码实践过程

实践一: 环境准备与数据创建

Shíjiàn yī: Huánjìng zhǔnbèi yǔ shùjù chuàngjiàn

1 打开 PyCharm，创建新项目

Dǎkāi PyCharm, chuàngjiàn xīn xiàngmù

> 👨‍🏫 **教授:** 同学们，请打开你们的 PyCharm。我们今天要创建一个新的项目(project)，名字叫 encoding_lab。

Tóngxuémen, qǐng dǎkāi nǐmen de PyCharm. Wǒmen jīntiān yào chuàngjiàn yígè xīn de xiàngmù, míngzì jiào encoding_lab.

> 👨‍🏫 **教授:** 然后在项目里创建一个新的 Python 文件(file)，文件名叫 lab01_encoding.py。跟我一起来!

Ránhou zài xiàngmù lǐ chuàngjiàn yígè xīn de Python wénjiàn, wénjiàn míng jiào lab01_encoding.py. Gēn wǒ yìqǐ lái!

2 安装必要的库 (library)

Ānzhuāng bìyào de kù

> 👨‍🏫 **教授:** 首先，我们需要安装一些必要的库(library)。请打开 PyCharm 的终端(Terminal)，输入以下命令(command):

Shǒuxiān, wǒmen xūyao ānzhuāng yìxiē bìyào de kù. Qǐng dǎkāi PyCharm de zhōngduān, shūrù yǐxià mìnglìng:

```
pip install pandas numpy scikit-learn
```

3 导入库并创建示例数据

Dǎorù kù bìng chuàngjiàn shìlì shùjù

> 👨‍🏫 **教授:** 好，现在我们开始写代码(code)。请跟我一起在 lab01_encoding.py 里输入以下内容:

```
# lab01_encoding.py
# 数据预处理 - 编码实践课程

import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, OneHotEncoder

# ====== 第1步: 创建示例数据 ======
data = {
    'name': ['张三', '李四', '王五', '赵六', '钱七', '孙八'],
    'gender': ['男', '女', '男', '女', '男', '女'],
    'education': ['本科', '研究生', '本科', '博士', '研究生', '本科'],
    'major': ['计算机', '数据科学', '人工智能',
```

```
            '计算机', '数据科学', '人工智能'],
    'grade': ['A', 'B', 'C', 'A', 'B', 'A']
}

df = pd.DataFrame(data)
print('原始数据:')
print(df)
print()
```

教授: 好，现在请大家点击运行按钮(Run button)，或者按 Shift+F10 运行代码。看看输出结果:

Hǎo, xiànzài qǐng dàjiā diǎnjī yùnxíng ànniǔ, huòzhě àn Shift+F10 yùnxíng dàimǎ. Kànkan shūchū jiéguǒ:

✓ 预期输出 (Expected Output):

```
原始数据:
  name gender education     major grade
0  张三      男        本科      计算机     A
1  李四      女       研究生    数据科学     B
2  王五      男        本科    人工智能     C
3  赵六      女        博士      计算机     A
4  钱七      男       研究生    数据科学     B
5  孙八      女        本科    人工智能     A
```

教授: 大家都看到这个表格了吗？这个数据里有很多文字(text)类型的列(column)，比如gender、education、major、grade。这些都是分类数据(categorical data)，计算机看不懂这些文字!

Dàjiā dōu kàndào zhège biǎogé le ma? Zhège shùjù lǐ yǒu hěnduo wénzì lèixíng de liè, bǐrú gender, education, major, grade. Zhèxiē dōu shì fēnlèi shùjù, jìsuànjī kànbudǒng zhèxiē wénzì!

学生: 老师，为什么计算机看不懂文字呢?

Lǎoshī, wèishénme jìsuànjī kànbudǒng wénzì ne?

教授: 因为机器学习模型(machine learning model)只能理解数字。就像计算器只能做加减乘除一样，不能直接计算“男”加“女”对吧？所以我们需要把文字变成数字，这就是编码(encoding)!

Yīnwèi jīqì xuéxí móxíng zhǐ néng lǐjiě shùzì. Jiù xiàng jìsuànqì zhǐ néng zuò jiā jiǎn chéng chú yíyàng, bù néng zhíjiē jìsuàn nán jiā nǚ duì ba? Suǒyǐ wǒmen xūyao bǎ wénzì biànchéng shùzì, zhè jiùshì biānmǎ!

实践二: 标签编码 (Label Encoding)

Shíjiàn èr: Biāoqiān biānmǎ

教授: 现在我们来学习第一种编码方法——标签编码(Label Encoding)。它的原理很简单，就是给每个类别分配一个数字。

Xiànzài wǒmen lái xuéxí dì yī zhǒng biānmǎ fāngfǎ — biāoqiān biānmǎ. Tā de yuánlǐ hěn jiǎndān, jiùshì gěi měi ge lèibié fēnpèi yígè shùzì.

1 性别(gender) 列的标签编码

教授: 请在代码后面继续输入:

```
# ====== 第2步: 标签编码 (Label Encoding) ======
print('=' * 50)
print('标签编码 (Label Encoding) 实践')
print('=' * 50)

# 创建 LabelEncoder 对象
le_gender = LabelEncoder()

# 对 gender 列进行标签编码
df['gender_encoded'] = le_gender.fit_transform(df['gender'])

print('\n性别编码结果:')
print(df[['name', 'gender', 'gender_encoded']])
print(f'\n编码映射关系: {dict(zip(le_gender.classes_,
le_gender.transform(le_gender.classes_)))}')
print()
```

✓ 预期输出:

```
==================================================
标签编码 (Label Encoding) 实践
==================================================

性别编码结果:
  name gender  gender_encoded
0  张三      男               1
1  李四      女               0
2  王五      男               1
3  赵六      女               0
4  钱七      男               1
5  孙八      女               0

编码映射关系: {'女': 0, '男': 1}
```

教授: 大家看，“女”变成了0，“男”变成了1。LabelEncoder 是按照字母顺序(alphabetical order)来分配数字的。“女”的拼音是 nǚ，“男”的拼音是 nán，所以 女=0，男=1。

Dàjiā kàn, nǚ biànchéng le 0, nán biànchéng le 1. LabelEncoder shì ànzhào zìmǔ shùnxù lái fēnpèi shùzì de.

2 学历(education) 列的标签编码

教授: 接下来我们对学历(education)列也做标签编码:

```
# 对 education 列进行标签编码
le_edu = LabelEncoder()
df['education_encoded'] = le_edu.fit_transform(df['education'])

print('学历编码结果:')
print(df[['name', 'education', 'education_encoded']])
print(f'编码映射: {dict(zip(le_edu.classes_, le_edu.transform(le_edu.classes_)))}')
print()
```

✓ 预期输出:

```
学历编码结果:
  name education  education_encoded
0  张三        本科                 1
1  李四       研究生                 2
2  王五        本科                 1
3  赵六        博士                 0
4  钱七       研究生                 2
5  孙八        本科                 1

编码映射: {'博士': 0, '本科': 1, '研究生': 2}
```

学生: 老师，这里博士=0，本科=1，研究生=2，但是学历应该是 本科 < 研究生 < 博士 呀!

这个顺序不对呀?

Lǎoshī, zhèlǐ bóshì=0, běnkē=1, yánjiūshēng=2, dànshì xuélì yīnggāi shì běnkē < yánjiūshēng < bóshì a! Zhège shùnxù búduì a?

教授: 问得非常好！这就是标签编码的一个重要缺点(diǎn)。它会让模型(model)误以为这些数字有大小关系。对于这种有顺序的数据，我们可以手动指定编码顺序!

Wèn de fēicháng hǎo! Zhè jiùshì biāoqiān biānmǎ de yígè zhòngyào quēdiǎn. Tā huì ràng móxíng wù yǐwéi zhèxiē shùzì yǒu dàxiǎo guānxì.

3 手动指定顺序编码 (Manual Ordinal Encoding)

Shǒudòng zhǐdìng shùnxù biānmǎ

```
# 手动指定顺序编码 - 本科 < 研究生 < 博士
edu_order = {'本科': 0, '研究生': 1, '博士': 2}
df['education_ordinal'] = df['education'].map(edu_order)

print('手动顺序编码结果:')
print(df[['name', 'education', 'education_encoded', 'education_ordinal']])
```

```
print()
```

✓ 预期输出:

```
手动顺序编码结果:
  name education  education_encoded  education_ordinal
0  张三        本科                 1                  0
1  李四       研究生                 2                  1
2  王五        本科                 1                  0
3  赵六        博士                 0                  2
4  钱七       研究生                 2                  1
5  孙八        本科                 1                  0
```

教授: 现在顺序对了！本科=0，研究生=1，博士=2。这样模型就能正确理解学历的顺序关系了！

Xiànzài shùnxù duì le! Běnkē=0, yánjiūshēng=1, bóshì=2. Zhèyàng móxíng jiù néng zhèngquè lǐjiě xuélì de shùnxù guānxì le!

⚠ 重要提示: 标签编码适合顺序型变量(ordinal variable)，但对名义型变量(nominal variable)可能产生问题！

实践三: 独热编码 (One-Hot Encoding)

Shíjiàn sān: Dúrè biānmǎ

教授: 接下来我们学习另一种非常重要的编码方法——独热编码(One-Hot Encoding)。它的原理就像开关(switch)一样，每个类别只有“开(1)”和“关(0)”两种状态。

Jiēxiàlái wǒmen xuéxí lìng yī zhǒng fēicháng zhòngyào de biānmǎ fāngfǎ — dúrè biānmǎ. Tā de yuánlǐ jiù xiàng kāiguān yíyàng, měi ge lèibié zhǐ yǒu kāi hé guān liǎng zhǒng zhuàngtài.

1 使用 pandas 的 get_dummies

教授: 我们先用 pandas 的 get_dummies 函数(function)，这是最简单的方法！请输入以下代码:

```
# ====== 第3步: 独热编码 (One-Hot Encoding) ======
print('=' * 50)
print('独热编码 (One-Hot Encoding) 实践')
print('=' * 50)

# 方法1: 使用 pandas 的 get_dummies
major_dummies = pd.get_dummies(df['major'], prefix='major')

print('\n专业(major)的独热编码结果:')
print(major_dummies)
print()
```

✓ 预期输出:

```
==================================================
独热编码 (One-Hot Encoding) 实践
==================================================

专业(major)的独热编码结果:
   major_人工智能  major_计算机  major_数据科学
0       False        True        False
1       False       False         True
2        True       False        False
3       False        True        False
4       False       False         True
5        True       False        False
```

教授: 大家看！原来一列 major 变成了三列！每个专业变成了一个单独的列。如果你是计算机专业，那么 major_计算机 这一列就是 True，其他列都是 False。

Dàjiā kàn! Yuánlái yī liè major biànchéng le sān liè! Měi ge zhuānyè biànchéng le yígè dāndú de liè.

2 使用 sklearn 的 OneHotEncoder

Shǐyòng sklearn de OneHotEncoder

教授: 现在我们用 sklearn 的 OneHotEncoder 来做同样的事情:

```
# 方法2: 使用 sklearn 的 OneHotEncoder
ohe = OneHotEncoder(sparse_output=False)
major_ohe = ohe.fit_transform(df[['major']])

print('sklearn OneHotEncoder 结果:')
print(f'类别名称: {ohe.categories_[0]}')
print(f'编码结果:\n{major_ohe}')
print()
```

✓ 预期输出:

```
sklearn OneHotEncoder 结果:
类别名称: ['人工智能' '计算机' '数据科学']
编码结果:
[[0. 1. 0.]
 [0. 0. 1.]
 [1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]
 [1. 0. 0.]]
```

教授: 看，结果是一样的！但是用了0和1来表示。比如第一行 [0. 1. 0.] 表示这个学生是计算机专业。

Kàn, jiéguǒ shì yíyàng de! Dànshì yòng le 0 hé 1 lái biǎoshì.

⚠ 重要提示: get_dummies 和 OneHotEncoder 都可以做独热编码。get_dummies 更简单，OneHotEncoder 更适合机器学习流水线(pipeline)！

实践四: 编码方法对比

Shíjiàn sì: Biānmǎ fāngfǎ duìbǐ

教授: 现在我们来做一个完整的对比，看看两种编码的区别。请输入以下代码：

```
# ====== 第4步: 编码方法对比 ======
print('=' * 50)
print('编码方法对比')
print('=' * 50)

# 对 grade 列同时应用两种编码
le_grade = LabelEncoder()
df['grade_label'] = le_grade.fit_transform(df['grade'])

grade_dummies = pd.get_dummies(df['grade'], prefix='grade')
df_compare = pd.concat([df[['name', 'grade', 'grade_label']], grade_dummies], axis=
1)

print('\n成绩编码对比:')
print(df_compare)
print()
```

✓ 预期输出:

```
==================================================
编码方法对比
==================================================

成绩编码对比:
  name grade  grade_label  grade_A  grade_B  grade_C
0  张三     A            0     True    False    False
1  李四     B            1    False     True    False
2  王五     C            2    False    False     True
3  赵六     A            0     True    False    False
4  钱七     B            1    False     True    False
5  孙八     A            0     True    False    False
```

教授: 大家注意看，标签编码只有一列，而独热编码有三列。这就是两种方法的核心区别！

Dàjiā zhùyì kàn, biāoqiān biānmǎ zhǐ yǒu yī liè, ér dúrè biānmǎ yǒu sān liè. Zhè jiùshì liǎng zhǒng fāngfǎ de héxīn qūbié!

对比表 (Comparison Table)

特征	标签编码 (Label)	独热编码 (One-Hot)
结果列数	1列	N列 (类别数量)
数据类型	整数 (0, 1, 2...)	二进制 (0 或 1)
适用场景	顺序型变量	名义型变量
优点	简单、内存少	无顺序假设

缺点	可能引入虚假顺序	维度增加

实践五: 综合练习

Shíjiàn wǔ: Zōnghé liànxí

教授: 最后，我们来做一个综合练习，把所有学到的编码方法一起应用。请创建新文件 lab02_complete.py:

Zuìhòu, wǒmen lái zuò yígè zōnghé liànxí, bǎ suǒyǒu xuédào de biānmǎ fāngfǎ yìqǐ yìngyòng.

```
# lab02_complete.py
# 综合练习: 完整的编码流程

import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, OneHotEncoder

# 创建更复杂的数据集
data = {
    'city': ['北京', '上海', '广州', '北京', '深圳', '上海',
             '广州', '深圳', '北京', '上海'],
    'weather': ['晴', '雨', '晴', '阴', '晴', '雨',
                '阴', '晴', '雨', '晴'],
    'temp_level': ['高', '中', '高', '低', '高', '中',
                   '低', '高', '中', '低'],
    'sales': [100, 80, 95, 60, 110, 75, 55, 105, 70, 50]
}

df = pd.DataFrame(data)
print('原始数据:')
print(df)
print()
```

教授: 运行后我们看到有城市(city)、天气(weather)、温度等级(temp_level)和销售额(sales)四列。现在我们要分析哪些列需要什么编码:

Yùnxíng hòu wǒmen kàndào yǒu chéngshì, tiānqì, wēndù děngjí hé xiāoshòué sì liè.

学生: city 和 weather 是名义型的，应该用独热编码； temp_level 有顺序，应该用顺序编码!

City hé weather shì míngyì xíng de, yīnggāi yòng dúrè biānmǎ; temp_level yǒu shùnxù, yīnggāi yòng shùnxù biānmǎ!

教授: 非常棒！说得完全正确！现在继续输入代码:

Fēicháng bàng! Shuō de wánquán zhèngquè!

```
# Step 1: 顺序编码 - 温度等级 (低 < 中 < 高)
temp_order = {'低': 0, '中': 1, '高': 2}
df['temp_encoded'] = df['temp_level'].map(temp_order)

# Step 2: 独热编码 - 城市和天气
df_encoded = pd.get_dummies(df, columns=['city', 'weather'],
                            prefix=['city', 'weather'])
```

```
# 删除原始的 temp_level 列
df_encoded = df_encoded.drop('temp_level', axis=1)

print('编码后的完整数据:')
print(df_encoded)
print(f'\n原始列数: {df.shape[1]}')
print(f'编码后列数: {df_encoded.shape[1]}')
```

✓ 预期输出:

```
编码后的完整数据:
   sales  temp_encoded  city_上海  city_北京  city_广州  city_深圳  weather_晴  weather_雨
weather_阴
0    100             2   False    True   False   False     True   False   False
1     80             1    True   False   False   False    False    True   False
2     95             2   False   False    True   False     True   False   False
...

原始列数: 4
编码后列数: 9
```

教授: 看，原来只有4列，编码后变成9列了！这就是独热编码的特点——列会变多。但是模型(model)可以更好地理解这些数据了！

Kàn, yuánlái zhǐ yǒu 4 liè, biānmǎ hòu biànchéng 9 liè le! Zhè jiùshì dúrè biānmǎ de tèdiǎn — liè huì biàn duō.

实践六: 逆向转换 (Inverse Transform)

Shíjiàn liù: Nìxiàng zhuǎnhuàn

> 🏫 **教授:** 有时候我们需要把数字变回文字，这就是逆向转换(inverse transform)。这在模型预测(prediction)之后解读结果时很有用!

Yǒu shíhou wǒmen xūyao bǎ shùzì biànhuí wénzì, zhè jiùshì nìxiàng zhuǎnhuàn.

```
# ====== 第5步: 逆向转换 ======
# 在 lab01_encoding.py 文件中继续

print('=' * 50)
print('逆向转换 (Inverse Transform)')
print('=' * 50)

# 假设模型预测结果是数字
predicted_values = [1, 0, 1, 0, 1]
original_labels = le_gender.inverse_transform(predicted_values)

print(f'\n数字值: {predicted_values}')
print(f'原始标签: {list(original_labels)}')
```

✓ 预期输出:

```
==================================================
逆向转换 (Inverse Transform)
==================================================

数字值: [1, 0, 1, 0, 1]
原始标签: ['男', '女', '男', '女', '男']
```

> 🏫 **教授:** 看，数字又变回文字了！1变回“男”，0变回“女”。这个功能在实际项目中非常实用!

Kàn, shùzì yòu biànhuí wénzì le! 1 biànhuí nán, 0 biànhuí nǚ.

课堂总结与课后练习

Kètáng zǒngjié yǔ kèhòu liànxí

☑ 今天学习的内容总结

Jīntiān xuéxí de nèiróng zǒngjié

编号	内容	方法	关键代码
1	数据创建	pandas DataFrame	`pd.DataFrame()`
2	标签编码	LabelEncoder	`fit_transform()`
3	顺序编码	map() + dict	`df.map(dict)`
4	独热编码	get_dummies / OHE	`pd.get_dummies()`
5	逆向转换	inverse_transform	`inverse_transform()`

✍ 课后练习 (Homework)

Kèhòu liànxí

> 🏫 **教授:** 今天的作业很简单，请大家创建一个新文件 homework_encoding.py，完成以下任务:

Jīntiān de zuòyè hěn jiǎndān, qǐng dàjiā chuàngjiàn yígè xīn wénjiàn homework_encoding.py, wánchéng yǐxià rènwù:

任务1: 创建以下数据并对所有分类列进行适当的编码。

Rènwù yī: Chuàngjiàn yǐxià shùjù bìng duì suǒyǒu fēnlèi liè jìnxíng shìdàng de biānmǎ.

```
# homework_encoding.py
# 请完成以下任务

data = {
    'fruit': ['苹果', '香蕉', '橘子', '苹果', '香蕉'],
    'size': ['大', '中', '小', '大', '小'],
    'color': ['红', '黄', '橙', '绿', '黄'],
    'price': [5.0, 3.0, 4.0, 5.5, 2.5]
}

# TODO: 请分析每列的类型，选择合适的编码方法
# 提示: fruit 和 color 是名义型 -> ?
#       size 是顺序型 -> ?
```

任务2: 对编码结果进行逆向转换，验证是否能恢复原始数据。

Rènwù èr: Duì biānmǎ jiéguǒ jìnxíng nìxiàng zhuǎnhuàn, yànzhèng shìfǒu néng huīfù yuánshǐ shùjù.

❓ 思考题 (Thinking Questions)

Sīkǎo tí

1. 如果一个分类变量有100个类别，使用独热编码会有什么问题?

Rúguǒ yígè fēnlèi biànliàng yǒu 100 ge lèibié, shǐyòng dúrè biānmǎ huì yǒu shénme wèntí?

2. 什么情况下应该用标签编码，什么情况下应该用独热编码?

Shénme qíngkuàng xià yīnggāi yòng biāoqiān biānmǎ, shénme qíngkuàng xià yīnggāi yòng dúrè biānmǎ?

3. sales 列需要编码吗? 为什么?

Sales liè xūyao biānmǎ ma? Wèishénme?

下节课预告: 字符串及文本处理实践 (String & Text Processing)

Xià jié kè yùgào: Zìfú chuàn jí wénběn chǔlǐ shíjiàn