机器学习-其他

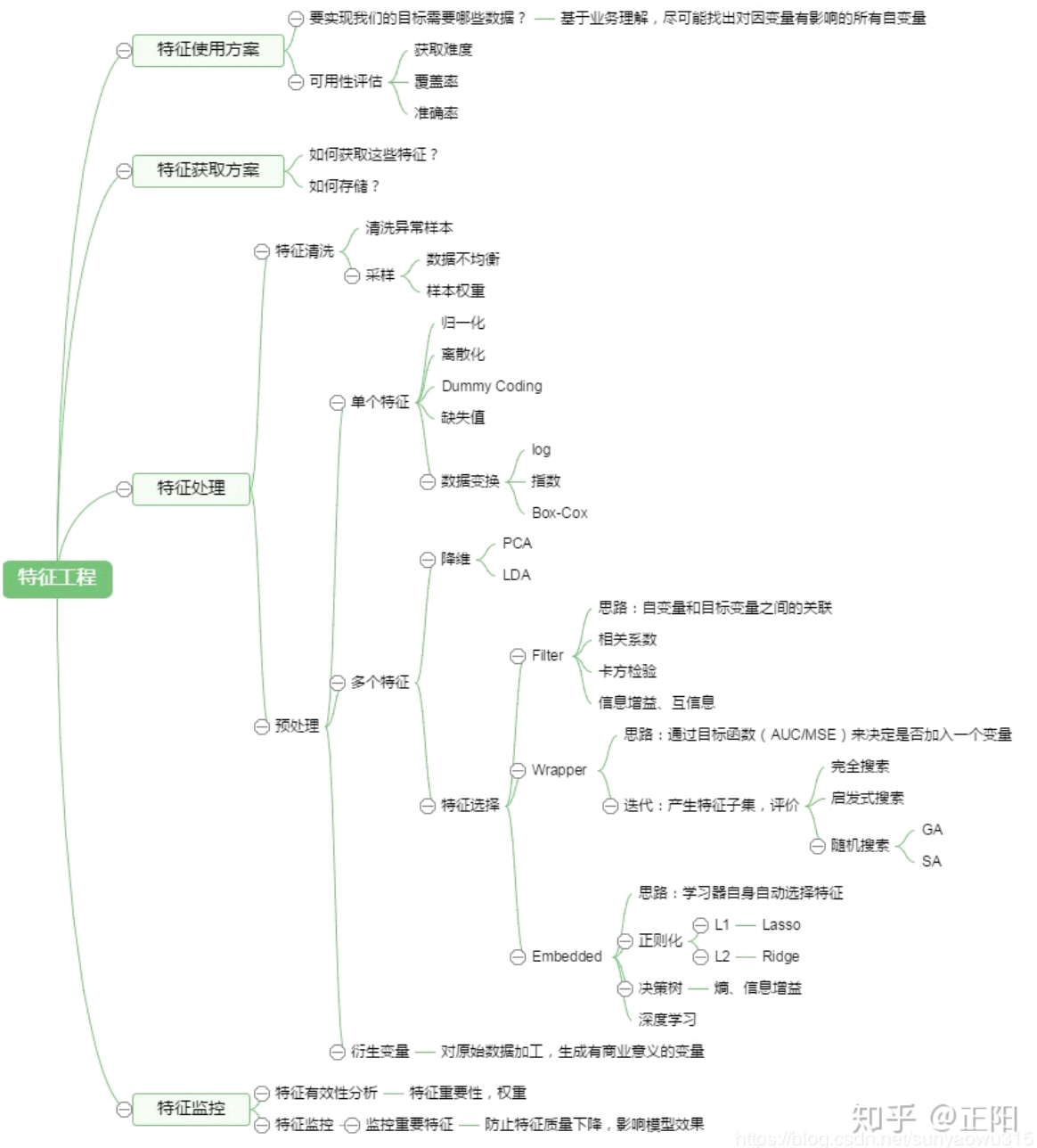
2021年4月5日　23:16

特征工程

参考：

《深度了解特征工程》很长，也很详细 https://zhuanlan.zhihu.com/p/111296130

《机器学习（19）——特征工程》 https://www.jianshu.com/p/ebc04e52d7c7



前言：特征工程是机器学习的重点，直接影响着模型的好坏。

数据收集

在进行机器学习之前，收集数据的过程中，我们主要按照以下规则找出我们所需要的数据：

1. 业务的实现需要哪些数据？ 基于对业务规则的理解，尽可能多的找出对因变量有影响的所有自变量数据。
2. 数据可用性评估在获取数据的过程中，首先需要考虑的是这个数据获取的成本；获取得到的数据，在使用之前，需要考虑一下这个数据是否覆盖了所有情况以及这个数 据的可信度情况。

- **数据源**
  用户行为日志数据：记录的用户在系统上所有操作所留下来的日志行为数据业务数据：
  商品/物品的信息、用户/会员的信息.....
  第三方数据：爬虫数据、购买的数据、合作方的数据....
- **数据储存**
  一般情况下，用于后期模型创建的数据都是存在在本地磁盘、关系型数据库或者一些相关的分布式数据存储平台的。 本地磁盘 MySQL Oracle HBase HDFS Hive

数据清洗

- **预处理**
  在数据预处理过程主要考虑两个方面，如下： 选择数据处理工具：关系型数据库或者Python 查看数据的元数据以及数据特征：一是查看元数据，包括字段解释、数据来源等一切可以描述数据的信息；另外是抽取一部分数据，通过人工查看的方式，对数据本身做一个比较直观的了解，并且初步发现一些问题，为之后的数据处理做准备。

- **格式内容错误数据清洗**
  时间、日期、数值、半全角等显示格式不一致：直接将数据转换为一类格式即可，该问 题一般出现在多个数据源整合的情况下。
  内容中有不该存在的字符：最典型的就是在头部、中间、尾部的空格等问题，这种情况下，需要以半自动校验加半人工方式来找出问题，并去除不需要的字符。
  内容与该字段应有的内容不符：比如姓名写成了性别、身份证号写成手机号等问题。

- **去除不需要的数据**
  一般情况下，我们会尽可能多的收集数据，但是不是所有的字段数据都是可以应用到模型构建过程的，也不是说将所有的字段属性都放到构建模型中，最终模型的效果就一定会好，实际上来讲，字段属性越多，模型的构建就会越慢，所以有时候可以考虑将不要的字段进行删除操作。在进行该过程的时候，要注意备份原始数据。

- **关联性验证**
  如果数据有多个来源，那么有必要进行关联性验证，该过程常应用到多数据源合并的过程中，通过验证数据之间的关联性来选择比较正确的特征属性，比如：汽车的线下购买信息和电话客服问卷信息，两者之间可以通过姓名和手机号进行关

联操作，匹配两者之间的车辆信息是否是同一辆，如果不是，那么就需要进行数据调整。

数据不平衡

- 设置损失函数的权重，使得少数类别数据判断错误的损失大于多数类别数据判断错误的损失，即当我们的少数类别数据预测错误的时候，会产生一个比较大的损失值，从而导致模型参数往让少数类别数据预测准确的方向偏。可以通过scikitlearn中的class_weight参数来设置权重。
- 下采样/欠采样(under sampling)：从多数类中随机抽取样本从而减少多数类别样本数据，使数据达到平衡的方式。
- 采用数据合成的方式生成更多的样本，该方式在小数据集场景下具有比较成功的案例。常见算法是SMOTE算法，该算法利用小众样本在特征空间的相似性来生成新样本。

特征转换

特征转换主要指将原始数据中的字段数据进行转换操作，从而得到适合进行算法模型构建的输入数据(数值型数据)，在这个过程中主要包括但不限于以下几种数据的处理：

- 文本数据转换为数值型数据
- 缺省值填充
- 定性特征属性哑编码
- 定量特征属性二值化
- 特征标准化与归一化

文本特征属性转换

机器学习的模型算法均要求输入的数据必须是数值型的，所以对于文本类型的特征属性，需要进行文本数据转换，也就是需要将文本数据转换为数值型数据。常用方式如下：

词袋法(BOW/TF)

TF-IDF(Term frequency-inverse document frequency)

HashTF

Word2Vec(主要用于单词的相似性考量)

缺省值填充

缺省值是数据中最常见的一个问题，处理缺省值有很多方式，主要包括以下四个步骤进行缺省值处理：

1. 确定缺省值范围
2. 去除不需要的字段
3. 填充缺省值内容
4. 重新获取数据

注意：最重要的是缺省值内容填充。

亚编码

亚编码(OneHotEncoder)：对于定性的数据(也就是分类的数据)，可以采用N位 的状态寄存器来对N个状态进行编码，每个状态都有一个独立的寄存器位，并且 在任意状态下只有一位有效；是一种常用的将特征数字化的方式。比如有一个特 征属性:['male','female']，那么male使用向量[1,0]表示，female使用[0,1]表示。

示例：

```
#-*- conding:utf-8 -*-
'''
亚编码，先把特征转为数字，再进行亚编码处理
'''
import numpy as np
import matplotlib as mpl
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import OneHotEncoder
path = "car.data"
data = pd.read_csv(path, header=None)
#查看数据
for i in range(7):
    print(np.unique(data[i]))
#将字符串转化为数字,用pd.Categorical
# pd.Categorical(data)
x=data.apply(lambda x:pd.Categorical(x).codes)
print(x.head(5))
#进行亚编码操作,toarray()方法转化为数组
en=OneHotEncoder()
x = en.fit_transform(x)
print(x.toarray())
print(en.n_values_)
#在进行转化
df=pd.DataFrame(x.toarray())
print(df.head(5))
```

二值化

二值化(Binarizer)：对于定量的数据（特征取值连续）根据给定的阈值，将其进行转换，如果大于阈值，那么赋值为1；否则赋值为0
注意：二值化非常常用，对每个特征按照不同阈值进行拆分，再进行合并
延伸为多值化，设定多个阈值。

标准化（z-score）
标准化：基于特征属性的数据(也就是特征矩阵的列)，获取均值和方差，然后将特征值转换至服从标准正态分布。计算公式如下：

$$x' = \frac{x - \overline{X}}{S}$$

区间缩放法
区间缩放法：是指按照数据的取值范围特性对数据进行缩放操作，将数据缩放到 给定区间上，常用的计算方式如下:

$$X\_std = \frac{X - X.\min}{X.\max - X.\min}$$

正则化
正则化：和标准化不同，正则化是基于矩阵的行进行数据处理，其目的是将矩阵的行均转换为“单位向量”，l2规则转换公式如下：

$$x' = \frac{x}{\sqrt{\sum_{j=1}^{m} x(j)^2}}$$

比较

标准化的目的是为了降低不同特征的不同范围的取值对于模型训练的影响；
比如对于同一个特征，不同的样本的取值可能会相差的非常大，那么这个时候一些异常小或者异常大的数据可能会误导模型的正确率；
另外如果数据在不同特征上的取值范围相差很大， 那么也有可能导致最终训练出来的模型偏向于取值范围大的特征，特别是在使用梯度下降求解的算法中；
通过改变数据的分布特征，具有以下两个好处：

1. 提高迭代求解的收敛速度；
2. 提高迭代求解的精度

归一化对于不同特征维度的伸缩变换的主要目的是为了使得不同维度度量之间特征具有 可比性，同时不改变原始数据的分布(相同特性的特征转换后，还是具有相同特性)。和标准化一样，也属于一种无量纲化的操作方式。

正则化则是通过范数规则来约束特征属性，通过正则化我们可以降低数据训练处来的模型的过拟合可能，和之前在机器学习中所讲述的L1、L2正则的效果一样。在进行正则化操作的过程中，不会改变数据的分布情况，但是会改变数据特征之间的相关特

性。
备注：广义上来讲，标准化、区间缩放法、正则化都是具有类似的功能。在有一些书籍 上，将标准化、区间缩放法统称为标准化，把正则化称为归一化操作。
简单来说
标准化会改变数据的分布情况，归一化不会，标准化的主要作用是提高迭代速度，降低不同维度之间影响权重不一致的问题。

增维

多项式扩展

多项式数据变换主要是指基于输入的特征数据按照既定的多项式规则构建更多的 输出特征属性，比如输入特征属性为[a,b]，当设置degree为2的时候，那么输出的多项式特征为[1, a, b, a^2, ab, b^2]

核函数

GBDT+LR

认为每个样本在决策树落在决策树的每个叶子上就表示属于一个类别， 那么我们可以进行基于GBDT或者随机森林的维度扩展，经常我们会将其 应用在GBDT将数据进行维度扩充， 然后使用LR进行数据预测，这也是我们进行所说的GBDT+LR做预测

降维

见《降维算法》的部分

特征选择

当做完特征转换后，实际上可能会存在很多的特征属性，比如：多项式扩展转换、文本数据转换等等，但是太多的特征属性的存在可能会导致模型构建效率降低，同时模型的效果有可能会变的不好，那么这个时候就需要从这些特征属性中选择出影响最大的特征属性作为最后构建模型的特征属性列表。

在选择模型的过程中，通常从两方面来选择特征：

- 特征是否发散：如果一个特征不发散，比如方差解决于0，也就是说这样的特征对于样本的区分没有什么作用。
- 特征与目标的相关性：如果与目标相关性比较高，应当优先选择。

特征选择的方法主要有以下三种：

- **Filter：过滤法**，按照发散性或者相关性对各个特征进行评分，设定阈值或者待选择阈值的个数，从而选择特征；常用方法包括方差选择法、相关系数法、卡方检验、 互信息法等。
- **Wrapper：包装法**，根据目标函数（通常是预测效果评分），每次选择若干特征或者排除若干特征；常用方法主要是递归特征消除法。
- **Embedded：嵌入法**，先使用某些机器学习的算法和模型进行训练，得到各个特征的权重系数，根据系数从大到小选择特征；常用方法主要是基于惩罚项的特征选择 法。

把以上进行一个汇总，如下图：

模型评价

参考：

《模型的评价方法》 https://zhuanlan.zhihu.com/p/53774460

评估方法

真实的数据不知道，那就从训练集中取出一部分数据当作测试集，其余当作训练集用来训练模型，对于测试集，套用模型后可以得到一个测试误差，我们就用测试误差去估计泛化误差。

此处的测试集合训练集是互斥的，即不能把训练过的数据又拿去当测试集，很显然，对于这类数据，会预测得很准确。

具体来说，对于数据集

$$D = \{(x_1, y_1), (x_2, y_2), \ldots (x_m y_m)\}$$

D是我们拿到的数据，同时要在D上进行训练和测试，因此要把D分成训练集S和测试集T，具体有以下几种做法。

留出法

显然，最直接的做法就是直接把D分成两个互斥的集合S和T，使得

$$S \cap T = \phi, S \cup T = D$$

这样一来，直接在S上训练模型，然后用T来检验模型的测试误差，进而估计模型的泛化能力

在sklearn中可以使用model_selection中的train_test_split来划分测试集和训练集，参数test_size表示测试集的比例。

```
from sklearn.model_selection import train_test_split
X_train,X_test, y_train, y_test =train_test_split(train_data,train_target,test_size=0.2, random_state=0)
```

当然，，默认情况下，sklearn中的划分方式是随机的，这种做法在数据量非常大时是问题不大的，但数据集不是很多的情况下，则还是需要分层抽样，保证S和T中各类样本的比例与原数据集D里面是一致的，这样的结果更具代表性，这个可以利用sklearn中的stratify来实现。

关于test_size的选择，理论上来说，我们是要用D中的数据来建模的，因此训练集占比越大，所建模型会越接近，但是此时会显得测试集数据过少，测试结果不具有普遍性。因此需要根据实际情况来选择，一般情况下会选择20%左右的数据作为测试集。

其实留出法的划分数据集的效果还是跟测试集的选取密切相关，为了降低其影响，可以选择多次划分数据集将最后结果取平均值的方式去处理。

交叉验证（详细见下一部分）

交叉验证的思想是先将数据集D划分为K个大小相等的互斥的子集(若没法均分，则前面几个子集样本数会多一个), 即

$$D = D_1 \cup D_1 \cup \ldots \cup D_k, D_i \cap D_j = \phi (i \neq j)$$

尽量保证每个子集中的数据的结构是一致的，每次选择1个子集作为测试集，剩余的k-1个子集重新求并集后作为训练集，共进行k次训练，返回k次训练的均值。

上式交叉验证的方法也叫“**K折交叉验证**”，显然，这种划分的结果与k的取值是密切相关的。

一般情况下，采用“10折交叉验证”。

```
from sklearn.model_selection import KFold
kf = KFold(n_splits=10)
```

与留出法类似，交叉验证同样存在划分的子集不一样的情况，因此也可以采取多次划分取平均值的方法。重复p次的k折交叉验证称为“**p次k折交叉验证**”。

当“k折交叉验证”中的k与样本数量m相等的时候，便成了一种特例，即每个样本均被划分成一份，取出一份作测试集，其余当作训练集，这种做法称为“**留一法**”。

自助法

当数据集较大时，常常采用的是留出法或者交叉验证法。但是当数据集较小时，还有一种自助法用来解决难以有效划分数据集的问题。

基本思路是：对于含有m个样本的数据集D，每次有放回地从其中取出一个样本，m次采样之后得到的D'，D中的样本在m次采样中均没有被取到的概率为

$$(1-\frac{1}{m})^m$$

取极限为1/e，约为0.368，这说明D中有大概36.8%的样本不在D'中。以D'为训练集，D|D'为测试集，则训练集中依然有着和原来数据一样多的样本，测试集的比例也有36.8%。这种方式在样本数据集太小，难以划分训练集、测试集时很有用，而且因为“新增”了样本，自助法在集成学习中帮助较大。但是，很明显这种做法改变了原有的数据集，引入了估计偏差，故实际中往往不采用。

我们在拿到含有m个样本的数据集D时，需要留出一部分数据作为测试集，因此实际训练的样本数目是小于m的，当验证完成，选定模型之后，还需要将原有的数据集D作为训练集再次进行训练，这样得到的模型才是最终需要的模型。

一般情况下，我们在实际过程中需要预测的数据成为测试集，因此上文中的数据集D划分出来的应该是“验证集”和训练集。通过验证集去选择模型、调整参数，算法、模型的泛化能力是需要在真正的测试集上进行比较的。

模型的性能度量

回归问题

常常采用的是均方误差（MSE）。

假设给定样本

$$D=\{(x_1,y_1),(x_2,y_2),\dots(x_m,y_m)\}$$

其中 y_i是x_i 的真实标记。

如果用模型f来预测D中的样本，则均方误差表示如下：

$$E(f;D)=\frac{1}{m}(f(x_i)-y_i)^2$$

如果D是连续集的话，p(x)为自变量x的概率密度函数，更一般的表达式如下：

$$E(f;D)=\int_D (f(x)-y)^2 p(x)dx$$

因此回归问题的度量标准较为简单，直接计算均方误差比较大小即可。

分类问题

有以下几种指标：

- 精度和错误率

对于分类问题，可以很直观的理解精度与错误率的概念。精度（accuracy）就是分类正确的样本数占总样本数的比例，其实也是正确率，而错误率就是分类错误的样本数占总样本数的比例，显然二者之和为1。
此处不再罗列其计算公式。

- 准确率和召回率

准确率（precision）也叫查准率，相比于上面的正确率来说是描述某一个类别的指标，而正确率是总体分类正确的比例。召回率（recall）也叫查全率，指的是某一类中用二分类的混淆矩阵来比较好理解

真实情况	预测结果	
	正例	反例
正例	TP (真正例)	FN (假反例)
反例	FP (假正例)	TN (真反例)

后续有关混淆矩阵、ROC和AUC的部分见《分类算法》最后评估的部分

交叉检验

参考：
《Cross-Validation（交叉验证）详解》 https://zhuanlan.zhihu.com/p/24825503
《机器学习之交叉验证》
https://blog.csdn.net/XiaoYi_Eric/article/details/86705101

- 交叉验证用在数据量不是很充足的情况(比如数据量小于一万条)，能够从有限的数据中获取尽可能多的有效信息。
- 交叉验证用于评估模型的预测性能，尤其是训练好的模型在新数据上的表现，能够一定程度上减小过拟合。

留出法

k折交叉验证

留一法交叉验证

此方法主要适用于数据量非常小的情况，比如N小于50的时候，推荐采用留一交叉验证