数据预处理

参考python机器学习之数据的预处理（五种方式数据处理案例详解）_python数据预处理-CSDN博客

1.归一化

在sklearn当中，我们使用preprocessing.MinMaxScaler来实现这个功能。MinMaxScaler有一个重要参数，feature_range，控制我们希望把数据压缩到的范围，默认是[0,1]。

$$x^* = \frac{x - min(x)}{max(x) - min(x)}$$

$$X_{std} = \frac{X - X.min(axis = 0)}{X.max(axis = 0) - X.min(axis = 0)}$$
$$X_{scaled} = X_{std} * (max - min) + min$$

X.min(axis=0)：每列中的最小值组成的行向量

X.max(axis=0)：每列中的最大值组成的行向量

max: 要映射到的区间最大值默认是1

min：要映射到的区间最小值，默认是0(data)

将标准化的数据映射到给定的[min,max]区间

```
scaler = MinMaxScaler(feature_range=[1,2])
print(scaler.fit_transform(data))
```

```
import numpy as np
X= np.array([[-1, 2], [-0.5, 6], [0, 10], [1, 18]])
X_nor= (X -  X.min(axis=0)) /(X.max(axis=0) - X.min(axis=0))
X_nor
```

2.标准化

当数据(x)按均值(μ)中心化后，再按标准差(σ)缩放，数据就会服从为均值为0，方差为1的正态分布（即标准正态分布），而这个过程，就叫做数据标准化(Standardization，又称Z-score normalization)

$$x^* = \frac{x - \mu}{\sigma}$$

```
scaler = StandardScaler()#实例化
scaler.fit_transform(data)#一步达成结果
```

标准化和归一化区别

大多数机器学习算法中，会选择StandardScaler来进行特征缩放，因为MinMaxScaler对异常值非常敏感。在PCA，聚类，逻辑回归，支持向量机，神经网络这些算法中StandardScaler往往是最好的选择。

MinMaxScaler在不涉及距离度量、梯度、协方差计算以及数据需要被压缩到特定区间时使用广泛，比如数字图像处理中量化像素强度时，都会使用MinMaxScaler将数据压缩于[0,1]区间之中。

建议先试试看StandardScaler，效果不好换MinMaxScaler。

3.缺失值填充

数据挖掘之中，常常会有重要的字段缺失值很多，但又不能舍弃字段的情况。因此，数据预处理中非常重要的一项就是处理缺失值。

- 使用，均值、中位数、0对数据进行填充
- 进行众数填充

```
data1.loc[:,"Age"] =
data1.loc[:,"Age"].fillna(data1.loc[:,"Age"].median())
#.fillna 在DataFrame里面直接进行填补

data1.dropna(axis=0,inplace=True)
#.dropna(axis=0)删除所有有缺失值的行，.dropna(axis=1)删除所有有缺失值的列
#参数inplace，为True表示在原数据集上进行修改，为False表示生成一个复制对象，
#不修改原数据，默认False
```

4.文本数据编码

在现实中，许多标签和特征在数据收集完毕的时候，都不是以数字来表现的。比如说，学历的取值可以是["小学"，"初中"，"高中"，"大学"]，付费方式可能包含["支付宝"，"现金"，"微信"]等等。在这种情况下，为了让数据适应算法和库，我们必须将数据进行编码，即是说，将文字型数据转换为数值型。

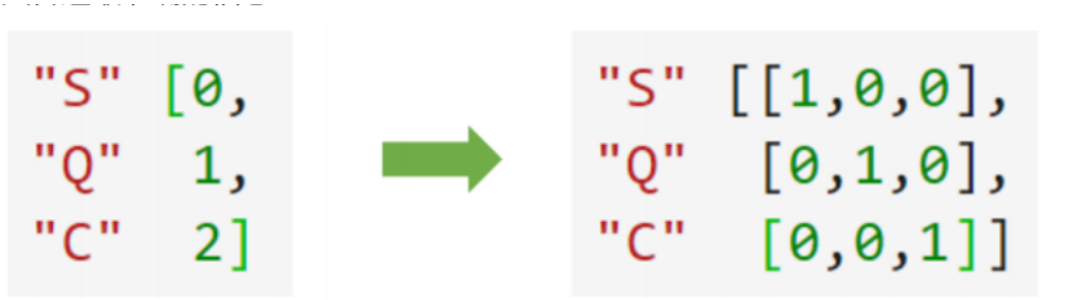
- 标签编码

  类别OrdinalEncoder可以用来处理有序变量，但对于名义变量，我们只有使用哑变量的方式来处理

- 特征编码

  独热编码，创建哑变量

  这样的变化，让算法能够彻底领悟，原来三个取值是没有可计算性质的，是"有你就没有我"的不等概念。在我们的 数据中，性别和舱门，都是这样的名义变量。因此我们需要使用独热编码，将两个特征都转换为哑变量。



5.处理连续型特征：二值化与分段

将连续特征值离散化

根据阈值将数据二值化（将特征值设置为0或1），用于处理连续型变量。大于阈值的值映射为1，而小于或等于阈值的值映射为0。默认阈值为0时，特征中所有的正值都映射到1。

6.异常值处理

- 异常值是指样本中的个别值，其数值明显偏离其余的观测值
- 异常值也称离群点，异常值的分析也称为离群点的分析
- 异常值分析 → 3σ原则 / 箱型图分析

- 异常值处理方法 → 删除 / 修正填补

  sklearn中的数据预处理和特征工程_sklearn.preprocessing库的作用-CSDN博客

机器学习基础|数据的降维及实战_降维数据输入时样本在x轴还是y轴-CSDN博客

7.数据降维

当我们拿到多特征的数据时，有一些字段的数据对于结果是没有意义，或者意义极小，但是在做机器学习的过程中也会参与计算，对我们最终分析结果造成不利影响，我们要根据实际情况，把数据进行降维，使得计算过程更轻便。

7.1 特征选择

特征选择是去除无关紧要或庸余的特征，仍然还保留其他原始特征，从而获得特征子集，从而以最小的性能损失更好地描述给出的问题。

特征选择方法可以分为三个系列：过滤式选择、包裹式选择和嵌入式选择的方法 。

7.1.1 过滤法 Filter

（1）方差过滤

通过特征本身的方差来筛选特征的类。比如一个特征本身的方差很小，就表示样本在这个特征上基本没有差异，可能特征中的大多数值都一样，甚至整个特征的取值都相同，那这个特征对于样本区分没有什么作用。所以无论接下来的特征工程要做什么，都要优先消除方差为0的特征。

```
select=VarianceThreshold(threshold=0)
#参数默认0
#表示方差的阈值，表示舍弃所有方差小于threshold的特征，不填默认为0，
#即删除所有的记录都相同的特征。
```

```
#将方差中位数作为参数threshold的值筛选出一半的特征
import numpy as np
import pandas as pd
#X.shape=(m,n)
X=pd.read_csv('data.csv')
X_fsvar=VarianceThreshold(np.median(X.var().values)).fit_transform(X)
#X_fsvar.shape=(m,n/2)
```

（2）相关性过滤

希望选出与标签相关且有意义的特征，因为这样的特征能够为我们提供大量信息。如果特征与标签无关，那只会白白浪费我们的计算内存，可能还会给模型带来噪音。 在sklearn当中，我们有三种常用的方法来评判特征与标签之间的相关性：卡方，F检验，互信息。

a、卡方过滤 卡方过滤是专门针对离散型标签（即分类问题）的相关性过滤。

卡方检验类 feature_selection.chi2 计算每个非负特征和标签之间的卡方统计量，并依照卡方统计量由高到低为特征排名。再结合 feature_selection.SelectKBest 这个可以输入"评分标准"来选出前K个分数最高的特征的类，我们可以借此除去最可能独立于标签，与我们分类目的无关的特征。

```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
chivalue,pvalues_chi = chi2(X_fsvar,y)
#chivalue表示卡方值
#pvalues_chi表示p值
#k值取，我们可以假定删除p值大于设定值的特征
k=chivalue.shape[0]-(pvalues_chi>0.05).sum()
#也可以直接取k=300，400等
X_fschi=SelectKBest(chi2,k).fit_transform(X_fsvar,y)
```

b、F检验 F检验，又称ANOVA，方差齐性检验，是用来捕捉每个特征与标签之间的线性关系的过滤方法。

F检验的本质是寻找两组数据之间的线性关系，其原假设是"数据不存在显著的线性关系"。

它返回F值和p值两个统计量。 p值小于0.05或0.01的特征： 这些特征与标签时显著线性相关的，应该保留。 p值大于0.05或0.01的特征： 和标签没有显著线性关系的特征，应该被删除。

```
from sklearn.feature_selection import f_classif
F,pvalues_f=f_classif(X_fsvar,y)
#F 表示F值
#pvalues_f 表示P值
k=F.shape[0]-(pvalues_f>0.05).sum()
X_fsF=SelectKBest(f_classif,k).fit_transform(X_fsvar,y)
```

c、互信息法 互信息法是用来捕捉每个特征与标签之间的任意关系（包括线性和非线性关系）的过滤方法

互信息法比F检验更加强大，F检验只能够找出线性关系，而互信息法可以找出任意关系。

互信息法不返回p值或F值类似的统计量，它返回"每个特征与目标之间的互信息量的估计"，这个估计量在[0,1]之间取值，为0则表示两个变量独立，为1则表示两个变量完全相关。

7.1.2 嵌入式选择

嵌入法是一种让算法自己决定使用哪些特征的方法，即特征选择和算法训练同时进行。

相比于过滤法，嵌入法的结果会更加精确到模型的效用本身，对于提高模型效力有更好的效果。并且，由于考虑特征对模型的贡献，因此无关的特征（需要相关性过滤的特征）和无区分度的特征（需要方差过滤的特征）都会因为缺乏对模型的贡献而被删除掉，可谓是过滤法的进化版

7.1.3 包装法 Wrapper

包装法也是一个特征选择和算法训练同时进行的方法，与嵌入法十分相似，它也是依赖于算法自身的选择

7.2 主成分分析

PCA是一种分析、简化数据集的技术，其核心在于数据维度压缩，尽可能降低原数据的维度，损失少量信息。 通过PCA可以实现削减回归分析或者聚类分析中特征的数量。在减少特征数量的同时尽可能减少信息的丢失。

当特征数量较少时是不推荐使用PCA的，比如当特征数量只有十几个二十几个，那就没必要了。只有当特征数量达到上百上千时，才考虑对数据进行简化。图片的特征数量通常上千，此时就适合做主成分分析，再比如分析巨量数据时，如医学领域，金融市场领域。

```
from sklearn.decomposition import PCA
def pca():
    pca = PCA(n_components=0.95)#设置为保留原数据的95%
    data = pca.fit_transform([[2,5,1,7],[5,1,7,8],[9,4,2,1]])
    print(data)
    return N
if __name__=="__main__":
   pca()
```

结果：

```
[[-2.08906587  4.21270874]
 [-3.73487921 -3.48737558]
 [ 5.82394508 -0.72533316]]
```

保存原数据95%的信息后，降成了2维。