



城东

一个懒惰的人，总是想设计更智能的程序来避免做重复性工作

996 人赞同了该回答

转自我的博文：

使用sklearn做单机特征工程

目录

1 特征工程是什么？

2 数据预处理

2.1 无量纲化

2.1.1 标准化

2.1.2 区间缩放法

2.1.3 标准化与归一化的区别

2.2 对定量特征二值化

2.3 对定性特征哑编码

2.4 缺失值计算

2.5 数据变换

3 特征选择

3.1 Filter

3.1.1 方差选择法

3.1.2 相关系数法

3.1.3 卡方检验

3.1.4 互信息法

3.2 Wrapper

3.2.1 递归特征消除法

3.3 Embedded

3.3.1 基于惩罚项的特征选择法

3.3.2 基于树模型的特征选择法

4 降维

4.1 主成分分析法（PCA）

4.2 线性判别分析法（LDA）

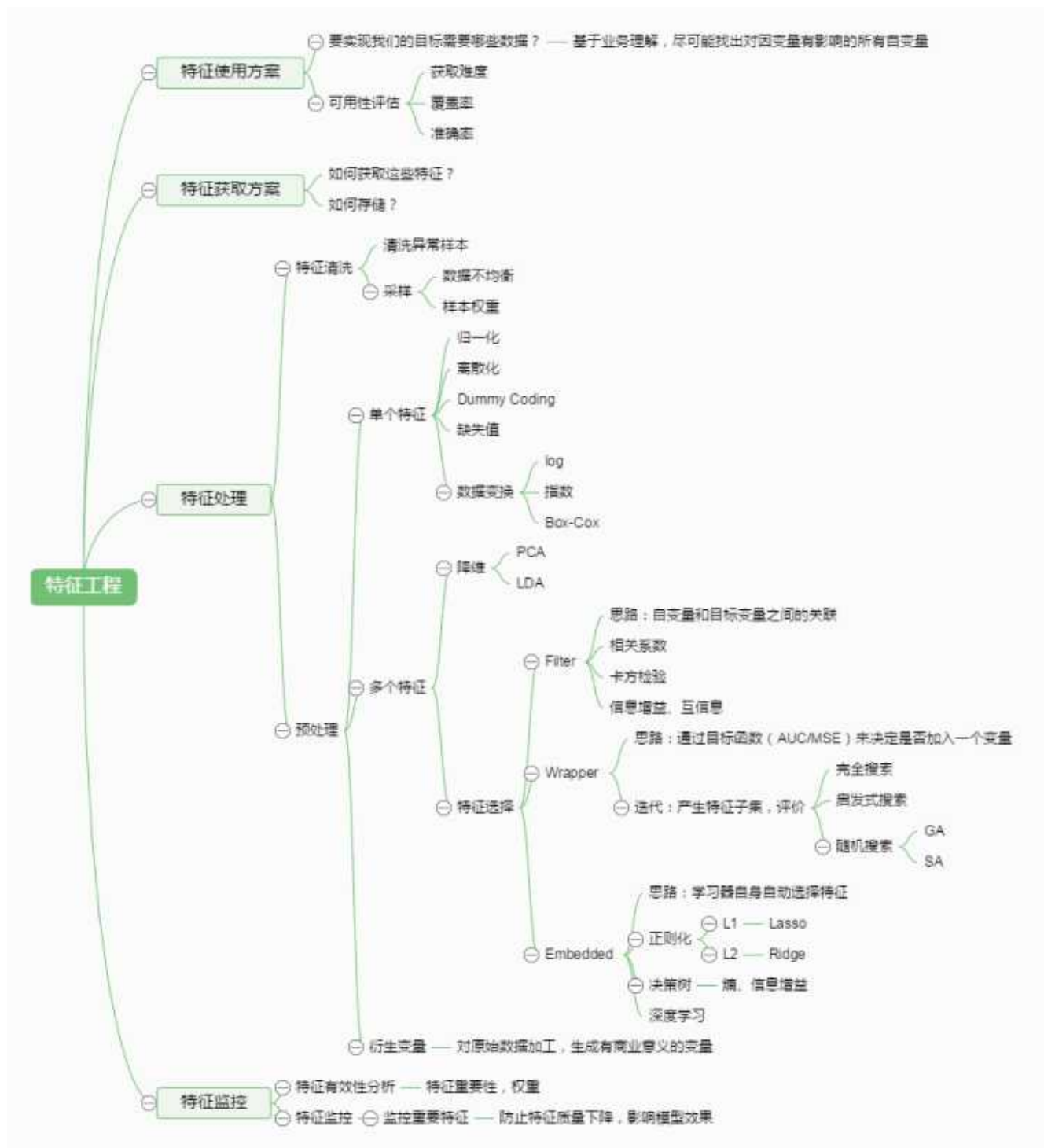
5 总结

6 参考资料

1 特征工程是什么？

有这么一句话在业界广泛流传：数据和特征决定了机器学习的上限，而模型和算法只是逼近这个上限而已。那特征工程到底是什么呢？顾名思义，其本质是一项工程活动，目的是最大限度地从原始数据中提取特征以供算法和模型使用。通过总结和归纳，人们认为特征工程包括

以下方面：



特征处理是特征工程的核心部分，sklearn提供了较为完整的特征处理方法，包括数据预处理，特征选择，降维等。首次接触到sklearn，通常会被其丰富且方便的算法模型库吸引，但是这里介绍的特征处理库也十分强大！

本文中使用sklearn中的IRIS（鸢尾花）数据集来对特征处理功能进行说明。IRIS数据集由Fisher在1936年整理，包含4个特征（Sepal.Length（花萼长度）、Sepal.Width（花萼宽度）、Petal.Length（花瓣长度）、Petal.Width（花瓣宽度）），特征值都为正浮点数，单位为厘米。目标值为鸢尾花的分类（Iris Setosa（山鸢尾）、Iris Versicolour（杂色鸢尾）），

Iris Virginica (维吉尼亚鸢尾))。导入IRIS数据集的代码如下：

```
from sklearn.datasets import load_iris

#导入IRIS数据集
iris = load_iris()

#特征矩阵
iris.data

#目标向量
iris.target
```

2 数据预处理

通过特征提取，我们能得到未经处理的特征，这时的特征可能有以下问题：

- 不属于同一量纲：即特征的规格不一样，不能够放在一起比较。无量纲化可以解决这一问题。
- 信息冗余：对于某些定量特征，其包含的有效信息为区间划分，例如学习成绩，假若只关心“及格”或不“及格”，那么需要将定量的考分，转换成“1”和“0”表示及格和未及格。二值化可以解决这一问题。
- 定性特征不能直接使用：某些机器学习算法和模型只能接受定量特征的输入，那么需要将定性特征转换为定量特征。最简单的方式是为每一种定性值指定一个定量值，但是这种方式过于灵活，增加了调参的工作。通常使用哑编码的方式将定性特征转换为定量特征：假设有N种定性值，则将这一个特征扩展为N种特征，当原始特征值为第i种定性值时，第i个扩展特征赋值为1，其他扩展特征赋值为0。哑编码的方式相比直接指定的方式，不用增加调参的工作，对于线性模型来说，使用哑编码后的特征可达到非线性的效果。
- 存在缺失值：缺失值需要补充。
- 信息利用率低：不同的机器学习算法和模型对数据中信息的利用是不同的，之前提到在线性模型中，使用对定性特征哑编码可以达到非线性的效果。类似地，对定量变量多项式化，或者进行其他的转换，都能达到非线性的效果。

我们使用sklearn中的preprocessing库来进行数据预处理，可以覆盖以上问题的解决方案。

2.1 无量纲化

无量纲化使不同规格的数据转换到同一规格。常见的无量纲化方法有标准化和区间缩放法。标准化的前提是特征值服从正态分布，标准化后，其转换成标准正态分布。区间缩放法利用了边界值信息，将特征的取值区间缩放到某个特点的范围，例如[0, 1]等。

2.1.1 标准化

标准化需要计算特征的均值和标准差，公式表达为：

使用preprocessing库的StandardScaler类对数据进行标准化的代码如下：

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

```
from sklearn.preprocessing import StandardScaler
```

```
#标准化，返回值为标准化后的数据
```

```
StandardScaler().fit_transform(iris.data)
```

2.1.2 区间缩放法

区间缩放法的思路有多种，常见的一种为利用两个最值进行缩放，公式表达为：

使用preprocessing库的MinMaxScaler类对数据进行区间缩放的代码如下：

$$x' = \frac{x - \text{Min}}{\text{Max} - \text{Min}}$$

```
from sklearn.preprocessing import MinMaxScaler
```

```
#区间缩放，返回值为缩放到[0, 1]区间的数据
```

```
MinMaxScaler().fit_transform(iris.data)
```

2.1.3 标准化与归一化的区别

简单来说，标准化是依照特征矩阵的列处理数据，其通过求z-score的方法，将样本的特征值转换到同一量纲下。归一化是依照特征矩阵的行处理数据，其目的在于样本向量在点乘运算或其他核函数计算相似性时，拥有统一的标准，也就是说都转化为“单位向量”。规则为L2的归一化公式如下：

$$x' = \frac{x}{\sqrt{\sum_j^m x[j]^2}}$$

使用preprocessing库的Normalizer类对数据进行归一化的代码如下：

```
from sklearn.preprocessing import Normalizer
```

```
#归一化，返回值为归一化后的数据
Normalizer().fit_transform(iris.data)
```

2.2 对定量特征二值化

定量特征二值化的核心在于设定一个阈值，大于阈值的赋值为1，小于等于阈值的赋值为0，公式表达如下：

$$x' = \begin{cases} 1, & x > threshold \\ 0, & x \leq threshold \end{cases}$$

使用preprocessing库的Binarizer类对数据进行二值化的代码如下：

```
from sklearn.preprocessing import Binarizer

#二值化，阈值设置为3，返回值为二值化后的数据
Binarizer(threshold=3).fit_transform(iris.data)
```

2.3 对定性特征哑编码

由于IRIS数据集的特征皆为定量特征，故使用其目标值进行哑编码（实际上是不需要的）。使用preprocessing库的OneHotEncoder类对数据进行哑编码的代码如下：

```
from sklearn.preprocessing import OneHotEncoder

#哑编码，对IRIS数据集的目标值，返回值为哑编码后的数据
OneHotEncoder().fit_transform(iris.target.reshape((-1,1)))
```

2.4 缺失值计算

由于IRIS数据集没有缺失值，故对数据集新增一个样本，4个特征均赋值为NaN，表示数据缺失。使用preprocessing库的Imputer类对数据进行缺失值计算的代码如下：

```
from numpy import vstack, array, nan
from sklearn.preprocessing import Imputer

#缺失值计算，返回值为计算缺失值后的数据
#参数missing_value为缺失值的表示形式，默认为NaN
#参数strategy为缺失值填充方式，默认为mean（均值）
Imputer().fit_transform(vstack((array([nan, nan, nan, nan]), iris.data)))
```

2.5 数据变换

常见的数据变换有基于多项式的、基于指数函数的、基于对数函数的。4个特征，度为2

的多项式转换公式如下：

$$(x'_1, x'_2, x'_3, x'_4, x'_5, x'_6, x'_7, x'_8, x'_9, x'_{10}, x'_{11}, x'_{12}, x'_{13}, x'_{14}, x'_{15}) \\ = (1, x_1, x_2, x_3, x_4, x_1^2, x_1 * x_2, x_1 * x_3, x_1 * x_4, x_2^2, x_2 * x_3, x_2 * x_4, x_3^2, x_3 * x_4, x_4^2)$$

使用preprocessing库的PolynomialFeatures类对数据进行多项式转换的代码如下：

```
from sklearn.preprocessing import PolynomialFeatures

#多项式转换
#参数degree为度，默认值为2
PolynomialFeatures().fit_transform(iris.data)
```

基于单变元函数的数据变换可以使用一个统一的方式完成，使用preprocessing库的FunctionTransformer对数据进行对数函数转换的代码如下：

```
from numpy import log1p
from sklearn.preprocessing import FunctionTransformer

#自定义转换函数为对数函数的数据变换
#第一个参数是单变元函数
FunctionTransformer(log1p).fit_transform(iris.data)
```

3 特征选择

当数据预处理完成后，我们需要选择有意义的特征输入机器学习的算法和模型进行训练。通常来说，从两个方面考虑来选择特征：

- 特征是否发散：如果一个特征不发散，例如方差接近于0，也就是说样本在这个特征上基本上没有差异，这个特征对于样本的区分并没有什么用。
- 特征与目标的相关性：这点比较显见，与目标相关性高的特征，应当优选选择。除方差法外，本文介绍的其他方法均从相关性考虑。

根据特征选择的形式又可以将特征选择方法分为3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，选择特征。
- 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。
- Embedded：集成法，先使用某些机器学习的算法和模型进行训练，得到各个特征的权值系数，根据系数从大到小选择特征。类似于Filter方法，但是是通过训练来确定特征的

优劣。

我们使用sklearn中的feature_selection库来进行特征选择。

3.1 Filter

3.1.1 方差选择法

使用方差选择法，先要计算各个特征的方差，然后根据阈值，选择方差大于阈值的特征。使用feature_selection库的VarianceThreshold类来选择特征的代码如下：

```
from sklearn.feature_selection import VarianceThreshold

#方差选择法，返回值为特征选择后的数据
#参数threshold为方差的阈值
VarianceThreshold(threshold=3).fit_transform(iris.data)
```

3.1.2 相关系数法

使用相关系数法，先要计算各个特征对目标值的相关系数以及相关系数的P值。用feature_selection库的SelectKBest类结合相关系数来选择特征的代码如下：

```
from sklearn.feature_selection import SelectKBest
from scipy.stats import pearsonr

#选择K个最好的特征，返回选择特征后的数据
#第一个参数为计算评估特征是否好的函数，该函数输入特征矩阵和目标向量，输出二元组（评分，
#参数k为选择的特征个数
SelectKBest(lambda X, Y: array(map(lambda x: pearsonr(x, Y), X.T)).T, k=2).fit_t
```

3.1.3 卡方检验

经典的卡方检验是检验定性自变量对定性因变量的相关性。假设自变量有N种取值，因变量有M种取值，考虑自变量等于i且因变量等于j的样本频数的观察值与期望的差距，构建统计量：

$$\chi^2 = \sum \frac{(A - E)^2}{E}$$

不难发现，这个统计量的含义简而言之就是自变量对因变量的相关性。用feature_selection库的SelectKBest类结合卡方检验来选择特征的代码如下：

```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
```


#选择K个最好的特征，返回选择特征后的数据

```
SelectKBest(chi2, k=2).fit_transform(iris.data, iris.target)
```

3.1.4 互信息法

经典的互信息也是评价定性自变量对定性因变量的相关性的，互信息计算公式如下：

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

为了处理定量数据，最大信息系数法被提出，使用feature_selection库的SelectKBest类结合最大信息系数法来选择特征的代码如下：

```
from sklearn.feature_selection import SelectKBest
from minepy import MINE
```

#由于MINE的设计不是函数式的，定义mic方法将其为函数式的，返回一个二元组，二元组的第2项

```
def mic(x, y):
    m = MINE()
    m.compute_score(x, y)
    return (m.mic(), 0.5)
```

#选择K个最好的特征，返回特征选择后的数据

```
SelectKBest(lambda X, Y: array(map(lambda x: mic(x, Y), X.T)).T, k=2).fit_transf
```

3.2 Wrapper

3.2.1 递归特征消除法

递归消除特征法使用一个基模型来进行多轮训练，每轮训练后，消除若干权值系数的特征，再基于新的特征集进行下一轮训练。使用feature_selection库的RFE类来选择特征的代码如下：

```
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression
```

#递归特征消除法，返回特征选择后的数据

#参数estimator为基模型

#参数n_features_to_select为选择的特征个数

```
RFE(estimator=LogisticRegression(), n_features_to_select=2).fit_transform(iris.
```

3.3 Embedded

3.3.1 基于惩罚项的特征选择法

使用带惩罚项的基模型，除了筛选出特征外，同时也进行了降维。使用feature_selection库的SelectFromModel类结合带L1惩罚项的逻辑回归模型，来选择特征的代码如下：

```
from sklearn.feature_selection import SelectFromModel
from sklearn.linear_model import LogisticRegression

#带L1惩罚项的逻辑回归作为基模型的特征选择
SelectFromModel(LogisticRegression(penalty="l1", C=0.1)).fit_transform(iris.dat
```

实际上，L1惩罚项降维的原理在于保留多个对目标值具有同等相关性的特征中的一个，所以没选到的特征不代表不重要。故，可结合L2惩罚项来优化。具体操作为：若一个特征在L1中的权值为1，选择在L2中权值差别不大且在L1中权值为0的特征构成同类集合，将这一集合中的特征平分L1中的权值，故需要构建一个新的逻辑回归模型：

```
from sklearn.linear_model import LogisticRegression

class LR(LogisticRegression):
    def __init__(self, threshold=0.01, dual=False, tol=1e-4, C=1.0,
                  fit_intercept=True, intercept_scaling=1, class_weight=None,
                  random_state=None, solver='liblinear', max_iter=100,
                  multi_class='ovr', verbose=0, warm_start=False, n_jobs=1):

        #权值相近的阈值
        self.threshold = threshold
        LogisticRegression.__init__(self, penalty='l1', dual=dual, tol=tol, C=C,
                                     fit_intercept=fit_intercept, intercept_scaling=intercept_scali
                                     random_state=random_state, solver=solver, max_iter=max_iter,
                                     multi_class=multi_class, verbose=verbose, warm_start=warm_star
        #使用同样的参数创建L2逻辑回归
        self.l2 = LogisticRegression(penalty='l2', dual=dual, tol=tol, C=C, fit

    def fit(self, X, y, sample_weight=None):
        #训练L1逻辑回归
        super(LR, self).fit(X, y, sample_weight=sample_weight)
        self.coef_old_ = self.coef_.copy()
        #训练L2逻辑回归
        self.l2.fit(X, y, sample_weight=sample_weight)

        cntOfRow, cntOfCol = self.coef_.shape
        #权值系数矩阵的行数对应目标值的种类数目
```

```

for i in range(cntOfRow):
    for j in range(cntOfCol):
        coef = self.coef_[i][j]
        #L1逻辑回归的权值系数不为0
        if coef != 0:
            idx = [j]
            #对应在L2逻辑回归中的权值系数
            coef1 = self.l2.coef_[i][j]
            for k in range(cntOfCol):
                coef2 = self.l2.coef_[i][k]
                #在L2逻辑回归中，权值系数之差小于设定的阈值，且在L1中对应的
                if abs(coef1-coef2) < self.threshold and j != k and sel
                    idx.append(k)
            #计算这一类特征的权值系数均值
            mean = coef / len(idx)
            self.coef_[i][idx] = mean
return self

```

使用feature_selection库的SelectFromModel类结合带L1以及L2惩罚项的逻辑回归模型，来选择特征的代码如下：

```

from sklearn.feature_selection import SelectFromModel

#带L1和L2惩罚项的逻辑回归作为基模型的特征选择
#参数threshold为权值系数之差的阈值
SelectFromModel(LR(threshold=0.5, C=0.1)).fit_transform(iris.data, iris.target)

```

3.3.2 基于树模型的特征选择法

树模型中GBDT也可用来作为基模型进行特征选择，使用feature_selection库的SelectFromModel类结合GBDT模型，来选择特征的代码如下：

```

from sklearn.feature_selection import SelectFromModel
from sklearn.ensemble import GradientBoostingClassifier

#GBDT作为基模型的特征选择
SelectFromModel(GradientBoostingClassifier()).fit_transform(iris.data, iris.tar

```

4 降维

当特征选择完成后，可以直接训练模型了，但是可能由于特征矩阵过大，导致计算量大，训练时间长的问题，因此降低特征矩阵维度也是必不可少的。常见的降维方法除了以上提到的基于L1惩罚项的模型以外，另外还有主成分分析法（PCA）和线性判别分析（LDA），线性判别分析本身也是一个分类模型。PCA和LDA有很多的相似点，其本质是要将原始的样本映射到维度更低的样本空间中，但是PCA和LDA的映射目标不一样：PCA是为了让映射后的样本具有最大的发散性；而LDA是为了让映射后的样本有最好的分类性能。所以说PCA是一种无监督的降维方法，而LDA是一种有监督的降维方法。

4.1 主成分分析法（PCA）

使用decomposition库的PCA类选择特征的代码如下：

```
from sklearn.decomposition import PCA

#主成分分析法，返回降维后的数据
#参数n_components为主成分数目
PCA(n_components=2).fit_transform(iris.data)
```

4.2 线性判别分析法（LDA）

使用lda库的LDA类选择特征的代码如下：

```
from sklearn lda import LDA

#线性判别分析法，返回降维后的数据
#参数n_components为降维后的维数
LDA(n_components=2).fit_transform(iris.data, iris.target)
```

5 总结

再让我们回归一下本文开始的特征工程的思维导图，我们可以使用sklearn完成几乎所有特征处理的工作，而且不管是数据预处理，还是特征选择，抑或降维，它们都是通过某个类的方法fit_transform完成的，fit_transform要不只带一个参数：特征矩阵，要不带两个参数：特征矩阵加目标向量。这些难道都是巧合吗？还是故意设计成这样？方法fit_transform中有fit这一单词，它和训练模型的fit方法有关联吗？接下来，我将在[《使用sklearn优雅地进行数据挖掘》](#)中阐述其中的奥妙！

6 参考资料

1. [FAQ: What is dummy coding?](#)

2. IRIS (鸢尾花) 数据集
3. 卡方检验
4. 干货：结合Scikit-learn介绍几种常用的特征选择方法
5. 机器学习中，有哪些特征选择的工程方法？
6. 机器学习中的数学(4)-线性判别分析 (LDA)，主成分分析(PCA)

编辑于 2016-07-19

▲996



● 40 条评论

➦ 分享

★ 收藏

♥ 感谢

收起▼



张戎

数学 话题的优秀回答者

130 人赞同了该回答

特征工程是一个非常重要的课题，是机器学习中不可缺少的一部分，但是它几乎很少出现于机器学习书本里面的某一章。在机器学习方面的成功很大程度上在于如果使用特征工程。

(I) 特征工程可以解决什么样的问题？

在机器学习中，经常是用一个预测模型（线性回归，逻辑回归，SVD等）和一堆原始数据来得到一些预测的结果，人们需要做的是从这堆原始数据中去提炼较优的结果，然后做到最优的预测。这个就包括两个方面，第一就是如何选择和使用各种模型，第二就是怎么样去使用这些原始的数据才能达到最优的效果。那么怎么样才能够获得最优的结果呢？贴上一句经典的话就是：

Actually the success of all Machine Learning algorithms depends on how you present the data.

----- Mohammad Pezeshki

直接翻译过来便是：事实上所有机器学习算法上面的成功都在于你怎么样去展示这些数据。由此可见特征工程在实际的机器学习中的重要性，从数据里面提取出来的特征好坏与否就会直接影响模型的效果。从某些层面上来说，所使用的特征越好，得到的效果就会越好。所需要的特征就是可以借此来描述已知数据的内在关系。总结一下就是：

Better feature means flexibility. Better feature means simpler models. Better feature means better results.

有的时候，可以使用一些不是最优的模型来训练数据，如果特征选择得好的话，依然可以得到一个不错的结果。很多机器学习的模型都能够从数据中选择出不错的结构，从而进行良好的预测。一个优秀的特征具有极强的灵活性，可以使用不那么复杂的，运算速度快，容易理解和维护的模型来得到不错的结果。

(II) 什么才是特征工程？

Feature Engineering is the process of transforming raw data into features that better

represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data.

----- Jason Brownlee

Feature Engineering is manually designing what the input x's should be.

----- Tomasz Malisiewicz

从这个概念可以看出，特征工程其实是一个如何展示和表现数据的问题，在实际工作中需要把数据以一种“良好”的方式展示出来，使得能够使用各种各样的机器学习模型来得到更好的效果。如何从原始数据中去除不佳的数据，展示合适的数据就成为了特征工程的关键问题。

（Ⅲ）特征有用性的预估

每次构造了一个特征，都需要从各个方面去证明该特征的有效性。一个特征是否重要主要在于该特征与要预测的东西是否是高度相关的，如果是高度相关，那么该特征就是十分重要的。比如常用的工具就是统计学里面的相关系数。

（Ⅳ）特征的构造过程

在实际工作中首先肯定要确定具体的问题，然后就是数据的选择和准备过程，再就是模型的准备和计算工作，最后才是展示数据的预测结果。构造特征的一般步骤：

[1] 任务的确定：根据具体的业务确定需要解决的问题；

[2] 数据的选择：整合数据，收集数据；

[3] 预处理数据：设计数据展现的格式，清洗数据，选择合适的样本使得机器学习模型能够使用它。比方说一些年龄特征是空值或者负数或者大于200等，或者说某个页面的播放数据大于曝光数据，这些就是数据的不合理，需要在使用之前把这一批数据排除掉。

[4] 特征的构造：转化数据，使之成为有效的特征。常用的方法是标准化，归一化，特征的离散化等。

（4.1）标准化（Standardization）：比方说有一些数字的单位是千克，有一些数字的单位是克，这个时候需要统一单位。如果没有标准化，两个变量混在一起搞，那么肯定就会不合适。

（4.2）归一化（Normalization）：归一化是因为在特征会在不同的尺度下有不同的表现形式，归一化会使得各个特征能够同时以恰当的方式表现。比方说某个专辑的点击播放率一般不会超过0.2，但是专辑的播放次数可能会达到几千次，所以说为了能够在模型里面得到更合适结果，需要先把一些特征在尺度上进行归一化，然后进行模型训练。

（4.3）特征的离散化（Discretization）：离散化是指把特征进行必要的离散处理，比方说年龄特征是一个连续的特征，但是把年龄层分成5 - 18岁（中小学生），19 - 23岁（大学生），24 - 29岁（工作前几年），30 - 40岁（成家立业），40 - 60岁（中年人）从某些层面来说比连续的年龄数据（比如说某人年龄是20岁1月3日之类的）更容易理解不同年龄层人的特性。典型的离散化步骤：对特征做排序 - > 选择合适的分割点 - > 作出区间的分割 - > 作出区间分割 - > 查看是否能够达到停止条件。

[5] 模型的使用：创造模型，选择合适的模型，用合适的模型来进行预测，用各种统计指标来判断该特征是否合适；

[6] 上线的效果：通过在线测试来看效果。

数据的转换（Transforming Data）就是把数据从原始的数据状态转换成适合模型计算的状态，从某些层面上来说，“数据转换”和“特征构造”的过程几乎是一致的。

（V）特征工程的迭代过程

特征工程的迭代步骤：

[1] 选择特征：需要进行头脑风暴（brainstorm）。通过具体的问题分析，查看大量的数据，从数据中查看出可以提取出数据的关键；

[2] 设计特征：这个需要具体问题具体分析，可以自动进行特征提取工作，也可以进行手工进行特征的构造工作，甚至混合两种方法；

[3] 选择特征：使用不同的特征构造方法，来从多个层面来判断这个特征的选择是否合适；

[4] 计算模型：通过模型计算得到模型在该特征上所提升的准确率。

[5] 上线测试：通过在线测试的效果来判断特征是否有效。

发布于 2016-01-23

▲130



● 13 条评论

➤ 分享

★ 收藏

♥ 感谢

收起▼



知乎用户

机器学习 话题的优秀回答者

63 人赞同了该回答

关于特征工程是什么这个问题，我强烈推荐读一篇文章：[Discover Feature Engineering, How to Engineer Features and How to Get Good at It.](#)

特征工程大概包括两个部分：特征提取和特征选择。关于特征选择可以参见这个问题：[机器学习中，有哪些特征选择的工程方法？ - 知乎用户的回答](#)。一般来说，领域内的知识主要是应用在特征提取。举个例子的话，比如说基于内容的购物推荐，性别就是一个很重要的领域知识，男性和女性关注的物品差别就比较大，推荐也应该体现出这种差别，那么这个特征就是这个问题一个重要特征，应该重点从已知数据提取，甚至专门为它再构建一个机器学习问题。

发布于 2015-04-05

▲63



● 添加评论

➤ 分享

★ 收藏

♥ 感谢



知乎用户

20 人赞同了该回答

水里没有鱼的情况下，再好的渔具也没用，特征工程就是找有鱼的那片水域。

编辑于 2016-02-06

**知乎用户**

相比于赞，更喜欢被评论

16 人赞同了该回答

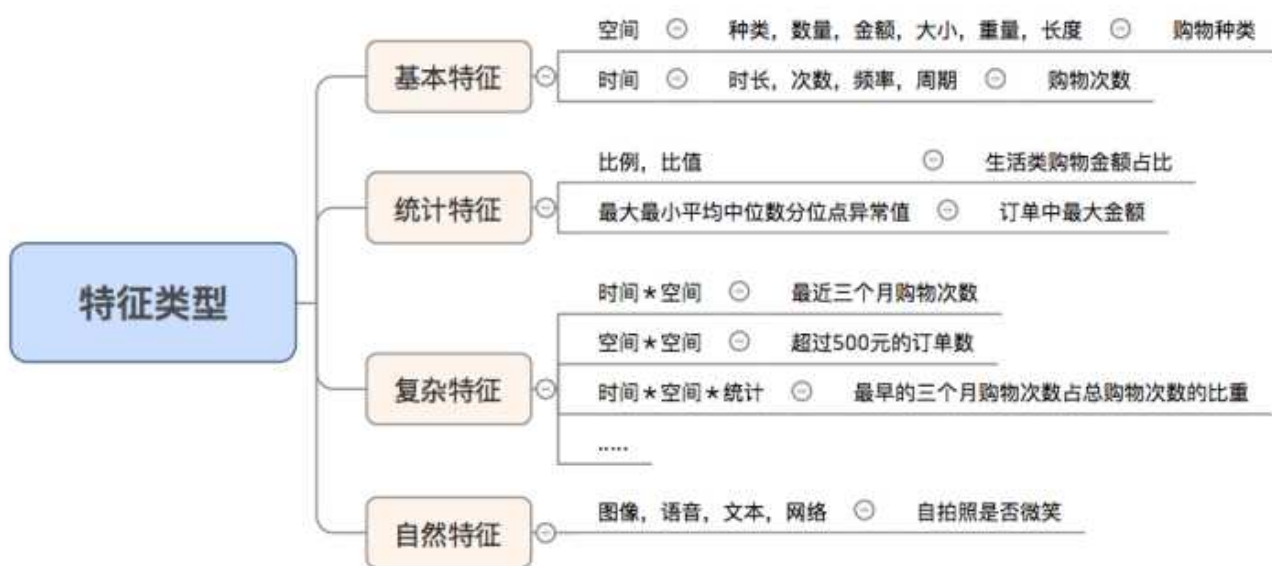
看到大家主要在谈数据的预处理和特征选择，很少讲到特征提取。

提取特征在不同领域差别较大，我尽可能抽象出一些通用的方法，总结了一点经验，希望大家交流~

如何提取特征？ - 知乎专栏

一.四类特征

我将特征分为四类：基本特征，复杂特征，统计特征和自然特征。



这里的特征设计主要针对非结构化数据，比如一个人的购物历史，运营商通话记录，位置的历史记录。当缺少领域知识无从下手的时候，可以从这四个方面来设计特征。

首先是基本特征，而后统计和复杂特征层层递进。其中针对图像语音等抽提特征有专用的知识方法，就不班门弄斧了。

掌握了这套特征设计的思路，在复杂数据上几乎可以设计出无穷无尽的特征。而怎么在最短的时间内，把数据中最有价值的特征提炼出来，就要考验数据挖掘工程师的功底。

二. 提取特征思路

基本思路是2/8法则，要花20%的时间，做50%的特征取得80%的效果。流程如下：

1. 做经验上最重要的
2. 做容易做的
3. 做可以批量产生的
4. 观察bad_case / good_feature,总结经验回到1

我主要是分析人的行为数据，很大程度上要依赖生活经验和领域知识。

将123步完成之后，通过数据去回归特征的效果，查看重要的特征效果是否符合预期，不符合的话为什么？怎么修改？哪些意料之外的特征很有效？对哪些人有效？为什么？

在不断的追问和查看数据的过程中，慢慢的积累领域知识和常识，形成自己的经验，形成对目标客群的深入认识。

三.杂谈

- 经验与偏见：

数据挖掘强调领域知识和常识经验。但是不要用经验生成一些很刻板的特征。比如要找犯罪分子，经验告诉你最近一个月刀具的人很可能是目标，制作特征的时候，不妨把最近1/3/6/12个月购买过道具 / 化学药品 / 火车票。。。这一类相关的特征都做出来，往往有新发现。要根据时间要求和对数据的熟悉程度，把握经验思路和刻画数据两者的平衡。

- 自动生成大量的特征：

当你的数据量足够大，可以尝试着设计规则，自动生成大量的特征。比如文本的one hot编码，高频电话通话次数。这种方式生成的特征往往会非常稀疏，对数据量的要求较高。数据不够经验来凑，对于大量稀疏的特征，又可以根据经验对特征分组合并来减少特征维度，提升模型效果。

- 空值，0值，特殊值（不存在的比例）：

对建模不熟悉的工程师经常会把异常的特征值都设置为0，或者把空值和0值混淆。一般来讲特征等于0是有特殊含义的，比如否或者数量上的0。对特征值拿不准的时候，最好不要拿0来填充。可以用一个特殊值比如 - 999来标示一些奇怪的特征取值（比如0/0），建模时候再做处理。实现特征提取的时候尽量不要留空值，这样一旦发现有特征是空值，就说明代码中有bug或者数据有异常，可以很方便的发现和处理边边角角的case。

- 复杂数据，根据相似性降维：

还有一种生成特征的思路很有意思。样本并没有明确的类别信息，但可以通过聚类将样本划归成几个类别，形成特征。这个思路在神经科学中就是spike sort，先将很多信号根据波形划分成几类（即信号来自于几个不同的细胞），而后再对每一个细胞产生的信号进行计算处理。

人或者轨迹同理，可以先通过降维聚类将相似的轨迹划分类别，再将类别作为特征进行处理。这样就将很复杂的数据整理成了有意义的特征。

编辑于 2017-03-17

▲16



3 条评论

分享

★收藏

♥感谢

收起



秋天的松鼠

码农/机器学习/健身/美食

11 人赞同了该回答

正好为了手头的项目在看特征工程的东西，仅从机器学习的角度说说我的个人理解。

对于机器学习来说，数据是算法的输入；要保证算法有好的表现，就要保证输入数据的质量。有时我们会遇到这样的情况：数据维数太高，甚至比数据点的个数还要多，这种情况下会对输入数据的质量造成以下影响：（1）维数冗余，有的数据只是噪声，完全可以去掉；（2）计算困难，维数的增加导致计算时间空间复杂度的增加；（3）容易过拟合，维数越高模型复杂度就可能越高。

这种情况下我们就需要特征工程来把数据“瘦身”成维数较少，维数质量高的数据集，去掉冗余信息。

通常来说常用的特征工程方法有特征提取和特征选择两种。特征提取主要通过PCA，CCA，ICA等方法从原来各维中总结出潜在的影响因子（latent features），就我的了解而言更注重数据变换，不太涉及领域内知识的运用（我的实践经验不多，如果其实是有的，还望指正）；而特征选择中可能涉及的领域知识（domain language）会更多一些。

特征选择是指直接从原有维数中选出有用的维来，一个常见的例子就是NLP中停用词（stop words）的使用。如果用一批文档的总词表（vocabulary）中的每一个单词作为一个维，去掉停用词就是去掉冗余维的一个体现。又比如之前做过fMRI脑成像的图像分类，领域内知识已知在某个实验里有效维的数据点会呈现一个特定的分布，我们就可以把完全不呈现这一分布的维剔除掉。当然特征选择也有不使用领域内知识的方法，比如使用L1 regularization自动进行特征选择等。

发布于 2015-12-07

▲11



添加评论

分享

★收藏

♥感谢



商行天下

数据挖掘、精准推荐、量化交易

8 人赞同了该回答

算法再牛逼，其上限也是由特征工程决定的。

发布于 2015-12-07

▲8



1 条评论

分享

★收藏

♥感谢



29 人赞同了该回答

科学总把简单的问题转化的很复杂，在彰显其严谨的同时，也把大部分的学习者挡在了门外，jacky跟大家谈谈如何深入浅出的学习特征工程？

《特征工程三部曲》之一数据处理

要理解特征工程，首先就要理解好数据（Data）和特征（Feature）的概念

（一）逻辑梳理

- 特征工程（Feature Engineering）
 - 其本质上是一项工程活动，它目的是最大限度地从原始数据中提取特征以供算法和模型使用。

特征工程在数据挖掘中有举足轻重的位置

数据领域一致认为：数据和特征决定了机器学习的上限，而模型和算法只能逼近这个上限而已。

- 特征工程重要性：
 - 特征越好，灵活性越强；
 - 特征越好，模型越简单；
 - 特征越好，性能越出色；

好特征即使使用一般的模型，也能得到很好的效果！好特征的灵活性在于它允许你可以选择不复杂的模型，同时，运行速度也更快，也更容易理解和维护。

好的特征，即使参数不是最优解，模型性能也能表现很好，因此，不需要太多时间去寻找最优参数，大大的降低了模型的复杂度，使模型趋向简单。

模型的性能包括模型的效果，执行的效率及模型的可解释性。特征工程的最终目的就是提升模型的性能。

数据科学家通过总结和归纳，把特征工程划分为以下三个部分：

- 特征工程包括：
 - 数据处理
 - 特征选择
 - 维度压缩

（二）数据处理

数据处理的常用技巧

- 量纲不一
- 虚拟变量
- 缺失值填充

1.数据处理——量纲不一

- 量纲：就是单位，特征的单位不一致，特征就不能放在一起比较。
- 解决量纲不一致的方法：标准化
 - 0-1标准化
 - Z标准化
 - Normalizer归一化

(1) 0-1标准化

是对原始数据进行线性变换，将特征值映射成区间为 [0 , 1] 的标准值中：

$$\text{标准化值} = \frac{\text{原数据} - \text{最小值}}{\text{最大值} - \text{最小值}}$$

(2) Z标准化

基于特征值的均值（ mean ）和标准差（ standard deviation ）进行数据的标准化。它的计算公式为：

$$\text{标准化数据} = \frac{\text{原数据} - \text{均值}}{\text{标准差}}$$

标准化后的变量值围绕0上下波动，大于0说明高于平均水平，小于0说明低于平均水平。

(3) Normalizer归一化

将每个样本缩放到单位范数（每个样本的范数为1），计算公式如下：

$$\bar{x} = \frac{x}{\sum_{i=1}^n x_i^2}$$

(4) 如何使用sklearn实现标准化

sklearn简介

- sklearn
 - 全名Scikit-Learn，是基于Python的机器学习模块，基于BSD开源许可证，官网上可以找到相关sklearn的资源，模块下载，文档，历程等等；
 - sklearn的数据结构基于numpy和pandas;
 - sklearn的数据计算基于scipy;
 - sklearn的数据可视化基于matplotlib;
- sklearn是在现有的数据分析，数据计算，数据可视化最好的包的基础上，搭建起来的最好python 机器学习的框架；
- sklearn的六大基本功能
 - 分类
 - 回归
 - 聚类
 - 数据降维
 - 模型选择
 - 模型预处理
- sklearn处理机器学习问题的三个步骤：
 - 数据准备与预处理
 - 模型选择与训练
 - 模型验证与参数调优

用sklearn实现标准化

```
#导入数据到data变量中
import pandas
data = pandas.read_csv('路径.csv')

# (一) Min-Max 标准化

from sklearn.preprocessing import MinMaxScaler
#初始化一个scaler对象
scaler = MinMaxScaler()
#调用scaler的fit_transform方法，把我们要处理的列作为参数传进去

data['标准化后的A列数据'] = scaler.fit_transform(data['A列数据'])
data['标准化后的B列数据'] = scaler.fit_transform(data['B列数据'])
```

(二) Z-Score标准化 (可在scale中直接实现)

```
from sklearn.preprocessing import scale
data['标准化后的A列数据'] = scale(data['A列数据'])
data['标准化后的B列数据'] = scale(data['B列数据'])
```

(三) Normalizer归一化

```
from sklearn.preprocessing import Normalizer
scaler = Normalizer()
#归一化可以同时处理多个列，所以[0]第一个进行赋值
data['归一化后的A列数据'] = scaler.fit_transform(data['A列数据'])[0]
data['归一化后的B列数据'] = scaler.fit_transform(data['B列数据'])[0]
```

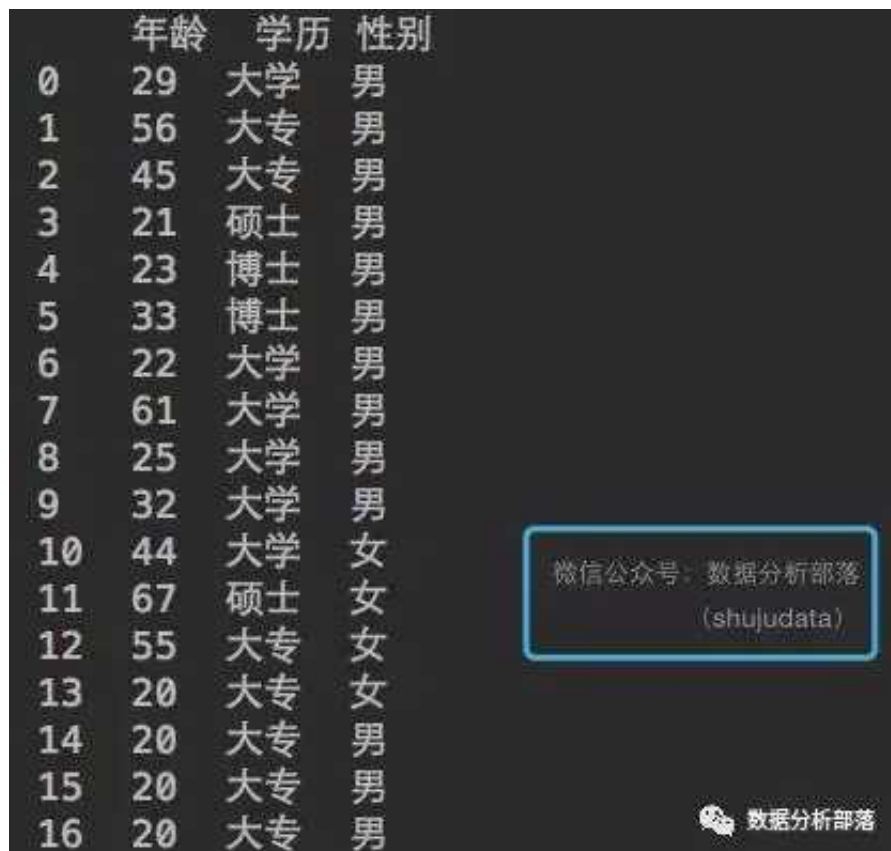
2.数据处理——虚拟变量

- 虚拟变量：也叫哑变量和离散特征编码，可用来表示分类变量、非数据因素可能产生的影响。
- 虚拟变量的两种数据类型：
 - 离散特征的取值之间有大小的意义：例如：尺寸 (L、XL、XXL)
 - 离散特征的取值之间没有大小的意义：例如：颜色 (Red、Blue、Green)
- 离散特征值有大小意义的虚拟变量处理
 - 离散特征的取值之间有大小意义的处理函数，我们只需要把大小值以字典的方式，作为第一个参数传入即可；
 - (1) dict 映射的字典
 - pandas.Series.map(dict)
- 离散特征值没有大小意义的虚拟变量处理
 - 离散特征的取值之间没有大小意义的处理方法，我们可以使用get_dummies方法处理，它有6个常用的参数
 - (1) data 要处理的DataFrame
 - (2) prefix 列名的前缀，在多个列有相同的离散项时候使用
 - (3) prefix_sep 前缀和离散值的分隔符，默认为下划线，默认即可
 - (4) dummy_na 是否把NA值，作为一个离散值进行处理，默认不处理
 - (5) columns 要处理的列名，如果不指定该列，那么默认处理所有列
 - (6) drop_first 是否从备选项中删第一个，建模的时候为避免共线性使用
 - pandas.getdummies(data,prefix=None,prefix_sep=' ',dummy_na=False,columns=None,drop_first=False)

虚拟变量——实战案例

以互联网金融行业为例：

```
import pandas
#有些朋友也可能是encoding='utf8'或其他
data=pandas.read_csv(
    'file:///Users/apple/Desktop/jacky_1.csv',encoding='GBK'
)
print(data)
```



	年龄	学历	性别
0	29	大学	男
1	56	大专	男
2	45	大专	男
3	21	硕士	男
4	23	博士	男
5	33	博士	男
6	22	大学	男
7	61	大学	男
8	25	大学	男
9	32	大学	男
10	44	大学	女
11	67	硕士	女
12	55	大专	女
13	20	大专	女
14	20	大专	男
15	20	大专	男
16	20	大专	男

其实，虚拟变量的实质就是要把离散型的数据转化为连续型的数据，因为第1列年龄已经是连续值，所以我们就不需要处理了。

我们看看如何处理学历和性别？

因为不同学历之间是有高低之分的，因此我们要使用Map方法来处理这种类型的离散型数据；

- 第1步：首先我们要处理不同学历之间的大小值
 - 我们使用drop_duplicates方法，来看看数据列都有哪些学历

#查看学历去重之后的情况

```
data['学历'].drop_duplicates()
```

- 第2步：理解数据值背后的意义，作出我们自己的解析，对每个学历进行评分


```
#构建学历字典
educationLevelDict={'博士':4,'硕士':3,'大学':2,'大专':1}
#调用Map方法进行虚拟变量的转换
data['Education Level Map']=data['Education Level'].map(educationLevelDict)
```

	年龄	学历	性别	学历 Map
0	29	大学	男	2
1	56	大专	男	1
2	45	大专	男	1
3	21	硕士	男	3
4	23	博士	男	4
5	33	博士	男	4
6	22	大学	男	2
7	61	大学	男	2
8	25	大学	男	2
9	32	大学	男	2
10	44	大学	女	2
11	67	硕士	女	3
12	55	大专	女	1
13	20	大专	女	1
14	20	大专	男	1
15	20	大专	男	1
16	20	大专	男	1

微信公众号：
数据分析部落
(shujudata)

数据分析部落

- 第3步 对于性别这种没有大小比较的离散变量，我们使用get_dummies方法，来进行调用处理即可；

```
dummies=pandas.get_dummies(
data,
columns=['性别'],
prefix=['性别'],
prefix_sep='_',
dummy_na=False,
drop_first=False)
```

完整代码展示

```
import pandas
data=pandas.read_csv(
'file:///Users/apple/Desktop/jacky_1.csv',encoding='GBK'
)
```

```

data['学历'].drop_duplicates()
educationLevelDict={'博士':4,'硕士':3,'大学':2,'大专':1}
data['学历 Map']=data['学历'].map(educationLevelDict)

dummies=pandas.get_dummies(
data,columns=['性别'],
prefix=['性别'],
prefix_sep='_',
dummy_na=False,
drop_first=False
)

print(dummies)

```

	年龄	学历	学历 Map	性别_女	性别_男
0	29	大学	2	0	1
1	56	大专	1	0	1
2	45	大专	1	0	1
3	21	硕士	3	0	1
4	23	博士	4	0	1
5	33	博士	4	0	1
6	22	大学	2	0	1
7	61	大学	2	0	1
8	25	大学	2	0	1
9	32	大学	2	0	1
10	44	大学	2	1	0
11	67	硕士	3	1	0
12	55	大专	1	1	0
13	20	大专	1	1	0
14	20	大专	1	0	1
15	20	大专	1	0	1
16	20	大专	1	0	1

微信公众号：
数据分析部落
(shujudata)



3.数据处理——缺失值填充

- 缺失值产生原因
 - 有些信息暂时无法获取；
 - 有些信息被遗漏或者错误的处理了
- 缺失值处理方法
 - 数据补齐

- 删除缺失行
- 不处理

实操 - 使用统计指标填充缺失值

```
import pandas
data=pandas.read_csv('路径.csv')
from sklearn.preprocessing import Imputer
# 'mean', 'median', 'most_frequent'

imputer=Imputer(strategy='mean')
imputer.fit_transform(data[['需填充列的列名']])
```

《特征工程三部曲》之二 数据选择

(一) 什么特征选择

- 特征选择 (Feature Selection)也称特征子集选择(Feature Subset Selection , FSS) , 或属性选择(Attribute Selection) , 是指从全部特征中选取一个特征子集, 使构造出来的模型更好。

(二) 为什么要做特征选择

- 在机器学习的实际应用中, 特征数量往往较多, 其中可能存在不相关的特征, 特征之间也可能存在相互依赖, 容易导致如下的后果:
 - 特征个数越多, 分析特征、训练模型所需的时间就越长。
 - 特征个数越多, 容易引起“维度灾难”, 模型也会越复杂, 其推广能力会下降。
- 特征选择能剔除不相关(irrelevant)或冗余(redundant)的特征, 从而达到减少特征个数, 提高模型精确度, 减少运行时间的目的。另一方面, 选取出真正相关的特征简化了模型, 使研究人员易于理解数据产生的过程。

(三) 特征选择基本原则

数据预处理完成之后, 我们需要选择有意义的特征, 输入机器学习的算法和模型进行训练, 通常来说, 从两个方面考虑来选择特征

- 如何选择特征
 - 是否发散
 - 是否相关

如果一个特征不发散, 例如方差接近于0, 也就是说样本在这个特征上基本没有差异, 那我

们就可以判断，这个特征对于样本的区别并没有什么用

第二个是特征与目标的相关性，与目标相关性高的特征应该优先选择

（四）特征选择常用的四种方法

我们以互联网金融实际情景举例：

	金融产品	累计销售（亿）	用户忠诚度评分
0	A产品	4.74	6.6
1	B产品	3.40	7.0
2	C产品	2.49	4.2
3	D产品	2.15	7.7
4	E产品	2.07	9.1

Process finished with exit code 0

微信公众号
数据分析部落
(shujudata)

数据分析部落

- 说白了，特征选择，就是看累计销售和用户忠诚度评分能不能成为金融产品的特征，我们会有一系统的评估标准，当然这些评估标准也都有人为主观判断在的。

1.方差选择法(过滤法)

- 使用方差选择法，先要计算各个特征的方差，然后根据阈值，选择方差大于阈值的特征。使用feature_selection库的VarianceThreshold类来选择特征的代码如下：

```
#首先把数据导入到data变量中
```

```
import pandas
```

```
data=pandas.read_csv('路径.csv')
```

```
#使用VarianceThreshold类进行方差过滤
```

```
from sklearn.feature_selection import VarianceThreshold
```

```
#要生成这个类的对象，就需要一个参数，就是最小方差的阈值，我们先设置为1，
```

```
#然后调用它的transform方法进行特征值的过滤
```

```
variancethreshold=VarianceThreshold(threshold=1)
```

```
variancethreshold.fit_transform(
```

```
data[['累计销售（亿）','用户忠诚度评分']]
```

```
)
```

```
#使用get_support方法，可以得到选择特征列的序号，
```

```
#然后根据这个序号在原始数据中把对应的列名选择出来即可
```

```
varianceThreshold.get_support()
```

threshold=1两个特征都被显示出来了

为什么阈值设定为1，累计销售与用户忠诚度这两个特征都被选择了出来？

首先我们看下累计销售与用户忠诚度各自的方差是什么？

```
#看下这两个特征的方差是什么？ data[['累计销售（亿）','用户忠诚度评分']].std()
```

```
累计销售（亿）      1.12145
用户忠诚度评分      1.79360
```

微信公众号：数据分析部落

2.相关系数法

- 先计算各个特征对目标值的相关系数，选择更加相关的特征。

以互联网金融行业为例

	月份	季度	广告推广费	注册并投资人数	销售金额
0	201601	1	29.4	14	800
1	201602	1	25.0	10	685
2	201603	1	19.5	7	620
3	201604	2	15.7	6	587
4	201605	2	9.0	5	565
5	201606	2	15.5	8	648
6	201607	3	10.6	6	593
7	201608	3	15.9	7	554
8	201609	3	17.0	7	600
9	201610	4	21.5	10	703
10	201611	4	22.6	11	728
11	201612	4	28.3	14	752
12	201701	1	24.9	14	765
13	201702	1	22.2	11	715
14	201703	1	17.0	8	601
15	201704	2	17.0	7	602

微信公众号：数据分析部落 (shujudata)

数据分析部落

```
#首先导入数据到data变量中
```

```
import pandas
```

```
data=pandas.read_csv('路径.csv')
```

```
#然后，SelectKBest类，通过回归的方法，以及要选择多少个特征值，
```

```
#新建一个 SelectKBest对象，
```

```
from sklearn.feature_selection import SelectKBest
```

```
from sklearn.feature_selection import f_regression
```

```
selectKBest = SelectKBest(
    f_regression,k=2
)
```

#接着，把自变量选择出来，然后调用`fit_transform`方法，

#把自变量和因变量传入，即可选出相关度最高的两个变量。

```
feature =data[['月份','季度','广告推广费','注册并投资人数']]
bestFeature =selectKBest.fit_transform(
    feature,
    data['销售金额']
)
```

#我们想要知道这两个自变量的名字，使用`get_support`方法即可得到相应的列名

```
feature.columns[selectKBest.get_support()]
```

最终，python帮助我们选择的特征是：

```
/Library/Frameworks/Python.framework/Versions/3.5/bin/python3.5
/Users/apple/PycharmProjects/untitled17/.ipynb_checkpoints/untitled17.ipy
Index(['广告推广费', '注册并投资人数'], dtype='object')
```

3.递归特征消除法

- 使用一个基模型来进行多轮训练，经过多轮训练后，保留指定的特征数。

还是延用上面那个案例

#首先导入数据到`data`变量中

```
import pandas
```

```
data=pandas.read_csv('路径.csv')
```

#接着，我们使用`RFE`类，在`estimator`中，

#把我们的基模型设置为线性回归模型`LinearRegression`，

#然后在把我们要选择的特征数设置为2，

#接着就可以使用这个`rfe`对象，把自变量和因变量传入`fit_transform`方法，

#即可得到我们需要的特征值

```
from sklearn.feature_selection import RFE
```

```
from sklearn.linear_model import LinearRegression
```

```
feature =data[['月份','季度','广告推广费','注册并投资人数']]
```

```

rfe =RFE(
    estimator=LinearRegression(),
    n_features_to_select=2
)
sFeature = rfe.fit_transform(
    feature,
    data['销售金额']
)

#同理，我们要想知道这两个自变量的名字，
#使用get_support方法，即可得到对应的列名
rfe.get_support()

```

4.模型选择法

- 它是一种我们把建好的模型对象传入选择器，然后它会根据这个已经建好的模型，自动帮我选择最好的特征值。

还是延用上面那个案例

```

import pandas
data = pandas.read_csv(
    'file:///Users/apple/Desktop/jacky_2.csv',
    encoding='GBK')

from sklearn.feature_selection import SelectFromModel
from sklearn.linear_model import LinearRegression

feature =data[['月份','季度','广告推广费','注册并投资人数']]
lrModel = LinearRegression()

selectFromModel = SelectFromModel(lrModel)

selectFromModel.fit_transform(
    feature,
    data['销售金额']
)
selectFromModel.get_support()

```

- 这里jacky就强调一点，模型选择法并不需要指定我们需要多少个特征，selectFromModel的方法会自动帮我们选择最优的特征数

(五) 总结

- 科学总把简单的问题转化的很复杂
- 有时间的小伙伴可以思考一下，特征选择的第一个方法对应的就是发散性，后面三个方法对应的就是回归。我们细细琢磨，后三个方法其实是一回事，但是得到的结果却略有不同，参透这其中的道理，也就参透数据挖掘和机器学习的奥秘了。

《特征工程三部曲》之三 维度压缩

当特征选择完成之后，就可以直接训练模型了，但是可能由于特征矩阵过大导致计算量大，训练时间长的的问题；因此，降低特征矩阵维度，也是必不可少的，主成分分析就是最常用的降维方法，在减少数据集的维度的同时，保持对方差贡献最大的特征，在sklearn中，我们使用PCA类进行主成分分析。

- 主成分分析 (Principal Components Analysis)
- PCA API
 - 有一个参数用于设置主成分的个数：`pca_3=PCA(n_components=3)`，设置好参数后，就可以生成PCA的对象了
 - 接着我们可以调用`fit_transform`方法对高维数据进行压缩：
`data_pca_3=pca3.fit_transform(data)`

我们人类只能看到三维数据，那么怎样把四维数据压缩到三维数据呢？

```
#导入iris特征数据到data变量中
import pandas
from sklearn import datasets
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from mpl_toolkits.mplot3d import Axes3D

iris = datasets.load_iris()

data = iris.data

#分类变量到target变量中
target = iris.target

#使用主成分分析，将四维数据压缩为三维
pca_3 = PCA(n_components=3)
data_pca_3 = pca_3.fit_transform(data)
```

```
#绘图
colors={0: 'r', 1: 'b', 2: 'k'}
markers={0: 'x', 1: 'D', 2: 'o'}

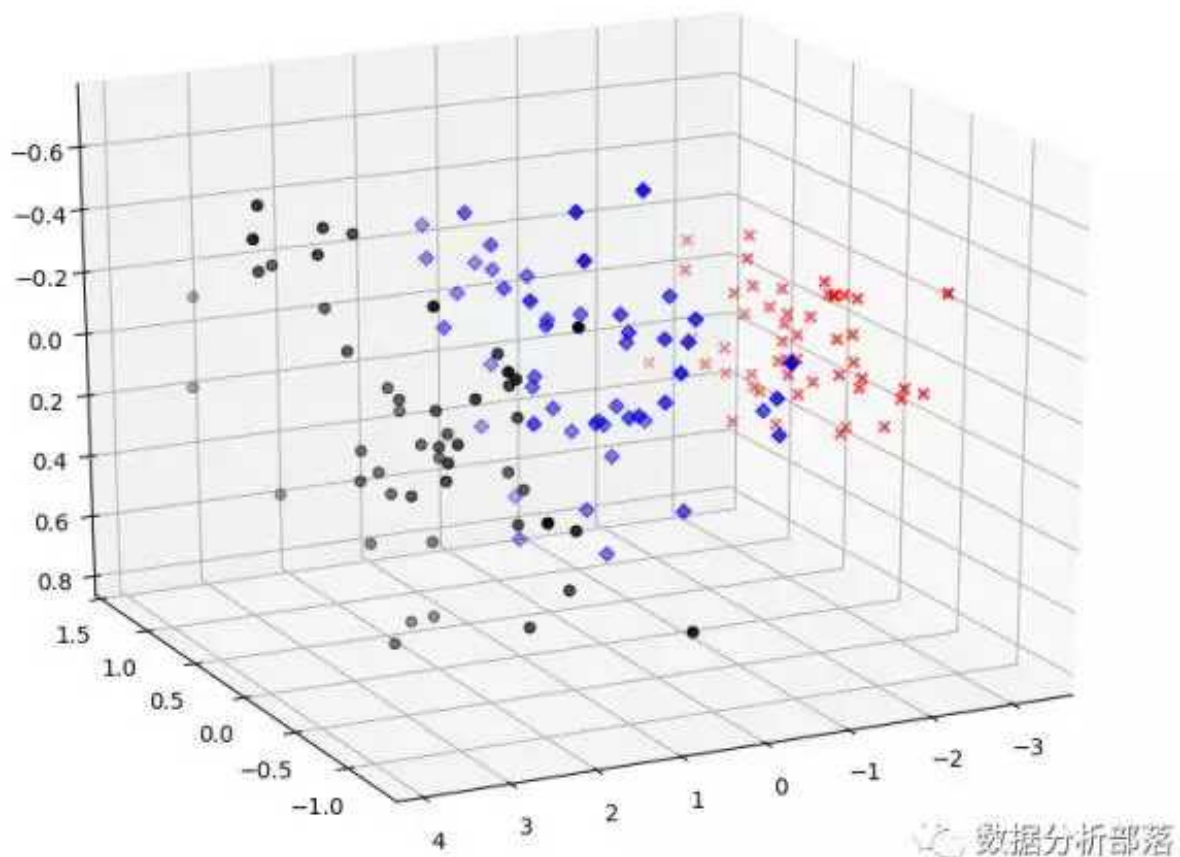
#弹出图形
%matplotlib qt

#三维数据
fig = plt.figure(1,figsize=(8,6))
ax = Axes3D(fig,elev=-150,azim=110)

data_pca_gb = pandas.DataFrame(
    data_pca_3
).groupby(target)

for g in data_pca_gb.groups:
    ax.scatter(
        data_pca_gb.get_group(g)[0],
        data_pca_gb.get_group(g)[1],
        data_pca_gb.get_group(g)[2],
        c=colors[g],
        marker=markers[g],
        cmap=plt.cm.Paired
    )
plt.show()
```

生成的效果图如下：



End

编辑于 2017-10-30

▲29



● 添加评论

➦ 分享

★ 收藏

♥ 感谢

收起▼



Eric D

Customer Insight / Data Analyst

3 人赞同了该回答

假如你会用机器学习算法，那么特征工程帮你选择最佳的学习材料，Data Engineer帮你整理好你需要的格式

发布于 2015-12-07

▲3



● 添加评论

➦ 分享

★ 收藏

♥ 感谢



辛俊波

机器学习/数据挖掘/户外/摄影

13 人赞同了该回答

看过一句话：一个团队，可以没有算法组，不能没有特征工程组

发布于 2015-04-20

▲13



💬 1 条评论

➦ 分享

★ 收藏

♥ 感谢



丁煌浩

慢点走，欣赏

8 人赞同了该回答

原始集：概率分布为A

训练集：概率分布为B

测试集：概率分布为C

特征工程：处理A，得到更接近C的B

机器学习：拟合B，用以预测C

发布于 2017-06-26

▲8



💬 添加评论

➦ 分享

★ 收藏

♥ 感谢



深圳吴彦祖

帅！

2 人赞同了该回答

大部分都是说常规机器学习的，要不我来说下**深度学习**的特征工程。

所谓特征工程，其实就是根据业务设计出合适的特征变量，做多了就知道这个东西**比模型重要多了**。数据质量高（指的是和业务有强关联）、特征工程好，随便整个算法效果也杠杠的，反之，调参什么的就很痛苦了，而且不一定有效。

那么，深度学习是怎么做特征工程呢？

其实，深度学习和一般机器学习最大的差别就是深度学习有进行“特征工程”的部分，所有的特征工程是它自己学的，而普通机器学习的特征组合是要你自己去**人肉发现并组合**的。



吴恩达的课有说一个经典的房价模型，假设我们就有下面几个特征：

1. 房屋大小
2. 卧室数量
3. 邮编
4. 富裕程度

你直接丢进模型效果肯定一般，要是哥，哥就这么做“特征工程”，我可以用“房屋大小” * “卧室数量”得到一个新特征，就叫他“家庭成员数”，这个有道理吧？然后“邮编” * “富裕程度”又是一个新特征，我们叫它“学校质量”，不同的邮编代表不同的地区，不同地区的教育水平肯定不一样，有钱与否决定了教育起点，这个特征肯定也是极好的。相信用这些新组成的特征和以前的特征效果肯定比只用基础特征好的多。

那么问题来了，人能发现多少特征组合呢？如果“特征工程”让模型自己去尝试，结果会好嘛？

答案是肯定的！

这就是为啥AlphaGo碾压人类而Alpha Zero碾压AlphaGo的原因——它站在一个我们没法想象的高度拿到了自己认为最屌的特征，这样得到新的特征后再进行其他步骤，自然事半功倍。

其实这这就是传说中的**表示学习**。

我们用CNN来“用人话”说明一下深度学习的“特征工程”是在干啥。

CNN有很多卷积层，比如我们现在要用CNN训练一个人脸识别模型，我们输入的特征都是像素点和RGB三个通道，神经网络会怎么处理这些特征呢。

在CNN前几层，表示的基本粒度由像素升级为较抽象的边(edges)，由于边有众多不同角度，如水平、垂直等，所以会分解(disentangle)为多个(边)特征(实际中远大于三)。而这层表示的基本粒度边(edges)是由多个明暗相间的像素经过组合变换得到。通过这些层的学习，你可

以想象简单的像素特征生成了“边缘”，就类似你画素描摹了个轮廓出来。

在中间几层依次类推，两条边交叉后组成了角corner, 多条边交叉后组成了contour, 而多个corners/ contours 又组成了人脸的众多部分，如鼻子、耳朵等等。这些组合成的鼻子耳朵也是模型自己学习到的特征。模型觉得我学到鼻子眼睛啥的就可以更好的帮我区分不同的人了，我们人类不也是如此咩。

最后，每一层的基本表示粒度都是由前一层多个元素组合而成。图片在每一层中都有新的表示，而第三个隐藏层的表示经过简单的线性分类器就能较好地预测图片的类别。

你看，前面那么多部分都是在做“特征工程”，做好特征工程吼，一个简单的线性分类器就可以很好的预测了。

不多BB了，应该是说明白了.....

发布于 2017-12-15

▲2 ▲ ● 添加评论 ➦ 分享 ★ 收藏 ♥ 感谢 收起▼



匿名用户

2 人赞同了该回答

我所理解的**特征工程**，应该是贯穿整个模型训练、预测以及后期应用的长期参考的工程，渗透在机器学习的整个过程。

比如NBA星探需要从CBA少年队选球员作为预备球员，现在NBA预备球员中缺少中锋，星探在找球员的时候可能就会注重球员的身高、篮下持球进攻能力，如果想找组织后卫，可能就更在乎球员的场上视野、运球、助攻等组织能力。最后他们组织的球队可能就是比较强的，因为方方面面都会考虑到。最后他们训练好这个选人的标准，球探们可能就会去欧洲，菲律宾等联赛里，按照这个标准去找人。这就是我理解的**特征工程**

到最后在进行不同的task的时候，特征工程就是一种经验，渗透在选模型、选参数、选特征表示等

编辑于 2016-04-15

▲2 ▲ ● 添加评论 ➦ 分享 ★ 收藏 ♥ 感谢



数据小虾米

公众号：数据小虾米 (DataShrimp)

1 人赞同了该回答

转自自己的公众号和博客。

对于机器学习应用而言，由于是用计算机强大的穷举能力去求解可能的解，除了算法设计与调优外，数据的获取、理解、选择才更是很多问题解决的核心。对此，业界的人起了个“特征工

程”的称呼来指代这个从数据转换为机器学习输入项的过程。

工程的关键是在给定实际局限的情况下给出满意的结果，之所以称为“特征工程”，或许也是因为从数据属性中抽取特征的过程更接近于一种界定于科学与实战之间的产物吧。它没有严格的数学理论证明为什么这样或那样选择或构造特征一定是好的，答案或许取决于“灵光乍现”，考虑实际应用各种限制，同时又不可能给予足够多的时间去穷尽所有的灵感，于是基于不同算法的前提假设，经验上归结出一些方法步骤出来，在这个意义上，称其为特征工程，其实更接近是一种经验艺术。

从机器学习的可学习理论出发，数据表达刻画的概念集是否针对问题足够完备，是否足够准确能提供对应算法的计算精度，是否足够丰富以保证学习模型能在偏差和方差间很好的平衡等等，都是特征工程需要重点考虑的问题。在这个意义上将，特征工程其实是一个复杂的系统工程。

在具体实践工作中，或可从以下五方面考虑：

特征于之前的数据特征不明显，通过一些数学的方法变换特征的值域空间，使得关注的特征值更显著。

- 特征构造。广义上讲，特征生成包括结合业务语义关联新的数据特征、或用算法构造新的训练数据特征等，狭义上也可包括综合两个或多个特征的交互作用而产生新的特征。

- 特征压缩。对于某些机器学习分类器而言，并不是越多的特征会产生更好的分类结果，去除一些非显著特征、噪音、异常点或更多特征细节对算法的影响。

- 特征缩放。包括把不同属性的特征，如分类特征、数值特征等变换为统一的格式，同时标准化还可以避免不同特征之间量纲不同的差异。

- 特征选取。相比其他方法，特征选取相对更成体系一些，目前已有Filter、Wrapper和Embedded三类方法，并且容易形成算法自动的执行。

特征工程可以认为是将大数据与人工智能连接在一起的桥梁，同时也是保证机器学习算法模型是否与业务目标问题匹配的关键制约因素。本文只是管中窥豹，试图从宏观角度做几笔简单的勾勒，今后有机会争取从更微观的问题出发写得更具体些。

编辑于 2017-01-12

▲1 ▲ ● 添加评论 ➤ 分享 ★ 收藏 ❤ 感谢



匿名用户

2 人赞同了该回答

。。。。。就是，大多数情况下，其实什么降噪什么分类的，随便选一个不太垃圾的就行，基本上指标上来全靠特征，说穿了就是人肉做一次分类，让机器学着舒服点。

个别公司的员工又来评论下面秀了，完全不知道他们公司是不是ppt驱动的，真心给跪，一点干货都没有，我也不和没有干货的人撕逼的，同学，回去写ppt吧，欢迎举报

当然，我在技术垃圾、产品也垃圾的百度，您可以认为大公司不这么做。

编辑于 2016-02-06

▲2 ▲ 18 条评论 分享 收藏 感谢



风间
准备养条狗

小tip，一般情况，一个机器学习任务给定label y ，如何收集数据设计特征提高拟合程度，正向思维是 $p(y|x)$ ，例如做房价预估，可能会想到地价，地价高，房价也高；遇到瓶颈时不妨逆向思考，找出 $p(x|y)$ ，上面例子，房价高，那么是不是地价也高

编辑于 2017-11-27

▲0 ▲ 添加评论 分享 收藏 感谢



霍某
哲学书

```
SelectKBest(lambda X, Y: array(map(lambda x: pearsonr(x, Y), X.T)).T, k=2).fit_t
```

发布于 2017-05-02

▲0 ▲ 1 条评论 分享 收藏 感谢



匿名用户

3 人赞同了该回答

都在瞎掰，特征工程说白了，就是根据很多其他表中的某些字段，进行运算，衍生出新的变量，比如：有一张登录表，记录每次登录的相关信息。然后衍生出一个变量，最近30天的登录次数

发布于 2017-01-24

▲3 ▲ 2 条评论 分享 收藏 感谢



本体匹配调谐
学会说话

一切唯有多参加比赛，多在现实问题历练！

发布于 2017-01-20

▲0



添加评论

分享

★收藏

♥感谢



首刀房主经验加倍

最简单的例子就是，如果给你10000列的数据，你先把他把变100列，再进行其他运算

发布于 2016-09-18

▲0



1 条评论

分享

★收藏

♥感谢