jasonfreak
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

使用**sklearn**优雅地进行数据挖掘

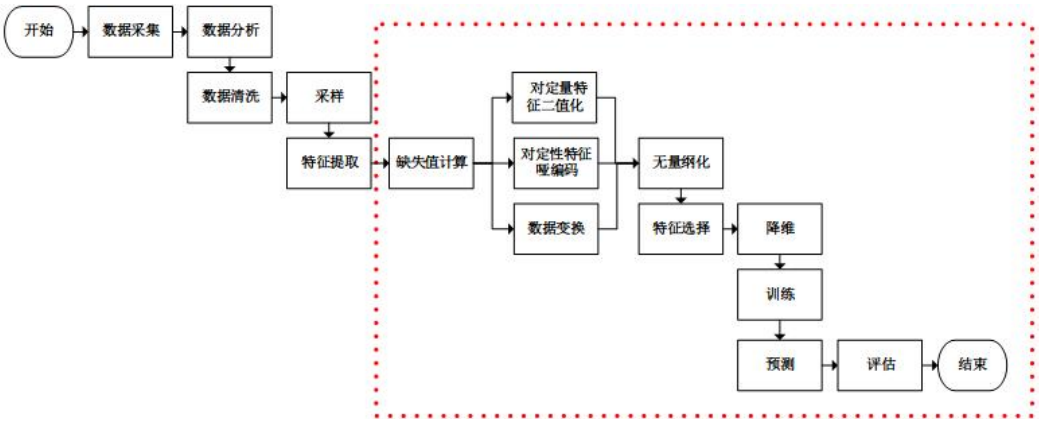
目录

1 使用sklearn进行数据挖掘

1.1 数据挖掘的步骤

数据挖掘通常包括数据采集，数据分析，特征工程，训练模型，模型评估等步骤。使用sklearn工具可以方便地进行特征工程和模型训练工作，在《使用sklearn做单机特征工程》中，我们最后留下了一些疑问：特征处理类都有三个方法fit、transform和fit_transform，fit方法居然和模型训练方法fit同名（不光同名，参数列表都一样），这难道都是巧合？

显然，这不是巧合，这正是sklearn的设计风格。我们能够更加优雅地使用sklearn进行特征工程和模型训练工作。此时，不妨从一个基本的数据挖掘场景入手：



我们使用sklearn进行虚线框内的工作（sklearn也可以进行文本特征提取）。通过分析sklearn源码，我们可以看到除训练，预测和评估以外，处理其他工作的类都实现了3个方法：fit、transform和fit_transform。从命名中可以看到，fit_transform方法是先调用fit然后调用transform，我们只需要关注fit方法和transform方法即可。

transform方法主要用来对特征进行转换。从可利用信息的角度来说，转换分为无信息转换和有信息转换。无信息转换是指不利用任何其他信息进行转换，比如指数、对数函数转换等。有信息转换从是否利用目标值向量又可分为无监督转换和有监督转换。无监督转换指只利用特征的统计信息的转换，统计信息包括均值、标准差、边界等等，比如标准化、PCA法降维等。有监督转换指既利用了特征信息又利用了目标值信息的转换，比如通过模型选择特征、LDA法降维等。通过总结常用的转换类，我们得到下表：

包	类	参数列表	类别	fit方法有用	说明
sklearn.preprocessing	StandardScaler	特征	无监督	Y	标准化
sklearn.preprocessing	MinMaxScaler	特征	无监督	Y	区间缩放

sklearn.preprocessing	Normalizer	特征	无信息	N	归一化
sklearn.preprocessing	Binarizer	特征	无信息	N	定量特征二值化
sklearn.preprocessing	OneHotEncoder	特征	无监督	Y	定性特征编码
sklearn.preprocessing	Imputer	特征	无监督	Y	缺失值计算
sklearn.preprocessing	PolynomialFeatures	特征	无信息	N	多项式变换（fit方法仅仅生成了多项式的表达式）
sklearn.preprocessing	FunctionTransformer	特征	无信息	N	自定义函数变换（自定义函数在transform方法中调用）
sklearn.feature_selection	VarianceThreshold	特征	无监督	Y	方差选择法
sklearn.feature_selection	SelectKBest	特征/特征+目标值	无监督/有监督	Y	自定义特征评分选择法
sklearn.feature_selection	SelectKBest+chi2	特征+目标值	有监督	Y	卡方检验选择法
sklearn.feature_selection	RFE	特征+目标值	有监督	Y	递归特征消除法
sklearn.feature_selection	SelectFromModel	特征+目标值	有监督	Y	自定义模型训练选择法
sklearn.decomposition	PCA	特征	无监督	Y	PCA降维
sklearn.lda	LDA	特征+目标值	有监督	Y	LDA降维

不难看到，只有有信息的转换类的fit方法才实际有用，显然fit方法的主要工作是获取特征信息和目标值信息，在这点上，fit方法和模型训练时的fit方法就能够联系在一起了：都是通过分析特征和目标值，提取有价值的信息，对于转换类来说是某些统计量，对于模型来说可能是特征的权值系数等。另外，只有有监督的转换类的fit和transform方法才需要特征和目标值两个参数。fit方法无用不代表其没实现，而是除合法性校验以外，其并没有对特征和目标值进行任何处理，Normalizer的fit方法实现如下：

```
1 def fit(self, X, y=None):
2         """Do nothing and return the estimator unchanged
3         This method is just there to implement the usual API and hence
4         work in pipelines.
5         """
6         X = check_array(X, accept_sparse='csr')
7         return self
```

基于这些特征处理工作都有共同的方法，那么试想可不可以将他们组合在一起？在本文假设的场景中，我们可以看到这些工作的组合形式有两种：流水线式和并行式。基于流水线组合的工作需要依次进行，前一个工作的输出是后一个工作的输入；基于并行式的工作可以同时进行，其使用同样的输入，所有工作完成后将各自的输出合并之后输出。sklearn提供了包pipeline来完成流水线式和并行式的工作。

1.2 数据初貌

在此，我们仍然使用IRIS数据集来进行说明。为了适应提出的场景，对原数据集需要稍微加工：

```
1 from numpy import hstack, vstack, array, median, nan
2 from numpy.random import choice
3 from sklearn.datasets import load_iris
4
5 #特征矩阵加工
6 #使用vstack增加一行含缺失值的样本(nan, nan, nan, nan)
7 #使用hstack增加一列表示花的颜色（0-白、1-黄、2-红），花的颜色是随机的，意味着颜色并不影响花的分类
8 iris.data = hstack((choice([0, 1, 2], size=iris.data.shape[0]+1).reshape(-1,1), vstack((iris.data,
array([nan, nan, nan, nan]).reshape(1,-1)))))
9 #目标值向量加工
10 #增加一个目标值，对应含缺失值的样本，值为众数
11 iris.target = hstack((iris.target, array([median(iris.target)])))
```

1.3 关键技术

并行处理，流水线处理，自动化调参，持久化是使用sklearn优雅地进行数据挖掘的核心。并行处理和流水线处理将多个特征

处理工作，甚至包括模型训练工作组合成一个工作（从代码的角度来说，即将多个对象组合成了一个对象）。在组合的前提下，自动化调参技术帮我们省去了人工调参的反锁。训练好的模型是贮存在内存中的数据，持久化能够将这些数据保存在文件系统中，之后使用时无需再进行训练，直接从文件系统中加载即可。

---

2 并行处理

并行处理使得多个特征处理工作能够并行地进行。根据对特征矩阵的读取方式不同，可分为整体并行处理和部分并行处理。整体并行处理，即并行处理的每个工作的输入都是特征矩阵的整体；部分并行处理，即可定义每个工作需要输入的特征矩阵的列。

2.1 整体并行处理

pipeline包提供了FeatureUnion类来进行整体并行处理：

```
from numpy import log1p
from sklearn.preprocessing import FunctionTransformer
from sklearn.preprocessing import Binarizer
from sklearn.pipeline import FeatureUnion

#新建将整体特征矩阵进行对数函数转换的对象
step2_1 = ('ToLog', FunctionTransformer(log1p))
#新建将整体特征矩阵进行二值化类的对象
step2_2 = ('ToBinary', Binarizer())
#新建整体并行处理对象
#该对象也有fit和transform方法，fit和transform方法均是并行地调用需要并行处理的对象的fit和transform方法
#参数transformer_list为需要并行处理的对象列表，该列表为二元组列表，第一元为对象的名称，第二元为对象
step2 = ('FeatureUnion', FeatureUnion(transformer_list=[step2_1, step2_2, step2_3]))
```

2.2 部分并行处理

整体并行处理有其缺陷，在一些场景下，我们只需要对特征矩阵的某些列进行转换，而不是所有列。pipeline并没有提供相应的类（仅OneHotEncoder类实现了该功能），需要我们在FeatureUnion的基础上进行优化：

View Code

在本文提出的场景中，我们对特征矩阵的第1列（花的颜色）进行定性特征编码，对第2、3、4列进行对数函数转换，对第5列进行定量特征二值化处理。使用FeatureUnionExt类进行部分并行处理的代码如下：

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

#新建将部分特征矩阵进行定性特征编码的对象
step2_1 = ('OneHotEncoder', OneHotEncoder(sparse=False))
#新建将部分特征矩阵进行对数函数转换的对象
step2_2 = ('ToLog', FunctionTransformer(log1p))
#新建将部分特征矩阵进行二值化类的对象
step2_3 = ('ToBinary', Binarizer())
#新建部分并行处理对象
#参数transformer_list为需要并行处理的对象列表，该列表为二元组列表，第一元为对象的名称，第二元为对象
#参数idx_list为相应的需要读取的特征矩阵的列
step2 = ('FeatureUnionExt', FeatureUnionExt(transformer_list=[step2_1, step2_2, step2_3],
idx_list=[[0], [1, 2, 3], [4]]))
```

---

3 流水线处理

pipeline包提供了Pipeline类来进行流水线处理。流水线上除最后一个工作以外，其他都要执行fit_transform方法，且上一个工作输出作为下一个工作的输入。最后一个工作必须实现fit方法，输入为上一个工作的输出；但是不限定一定有transform方法，因为流水线的最后一个工作可能是训练！

根据本文提出的场景，结合并行处理，构建完整的流水线的代码如下：

```
from numpy import log1p
from sklearn.preprocessing import Imputer
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import FunctionTransformer
from sklearn.preprocessing import Binarizer
from sklearn.preprocessing import MinMaxScaler
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
from sklearn.decomposition import PCA
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline

#新建计算缺失值的对象
step1 = ('Imputer', Imputer())
#新建将部分特征矩阵进行定性特征编码的对象
step2_1 = ('OneHotEncoder', OneHotEncoder(sparse=False))
#新建将部分特征矩阵进行对数函数转换的对象
step2_2 = ('ToLog', FunctionTransformer(log1p))
#新建将部分特征矩阵进行二值化类的对象
step2_3 = ('ToBinary', Binarizer())
#新建部分并行处理对象，返回值为每个并行工作的输出的合并
step2 = ('FeatureUnionExt', FeatureUnionExt(transformer_list=[step2_1, step2_2, step2_3],
idx_list=[[0], [1, 2, 3], [4]]))
#新建无量纲化对象
step3 = ('MinMaxScaler', MinMaxScaler())
#新建卡方校验选择特征的对象
step4 = ('SelectKBest', SelectKBest(chi2, k=3))
#新建PCA降维的对象
step5 = ('PCA', PCA(n_components=2))
#新建逻辑回归的对象，其为待训练的模型作为流水线的最后一步
step6 = ('LogisticRegression', LogisticRegression(penalty='l2'))
#新建流水线处理对象
#参数steps为需要流水线处理的对象列表，该列表为二元组列表，第一元为对象的名称，第二元为对象
pipeline = Pipeline(steps=[step1, step2, step3, step4, step5, step6])
```

4 自动化调参

网格搜索为自动化调参的常见技术之一，grid_search包提供了自动化调参的工具，包括GridSearchCV类。对组合好的对象进行训练以及调参的代码如下：

```
from sklearn.grid_search import GridSearchCV

#新建网格搜索对象
#第一参数为待训练的模型
 #param_grid为待调参数组成的网格，字典格式，键为参数名称（格式“对象名称__子对象名称__参数名称”），值为可取的参数值列表
 grid_search = GridSearchCV(pipeline, param_grid={'FeatureUnionExt__ToBinary__threshold':[1.0, 2.0,
3.0, 4.0], 'LogisticRegression__C':[0.1, 0.2, 0.4, 0.8]})
#训练以及调参
grid_search.fit(iris.data, iris.target)
```

5 持久化

externals.joblib包提供了dump和load方法来持久化和加载内存数据：

```
#持久化数据
#第一个参数为内存中的对象
#第二个参数为保存在文件系统中的名称
#第三个参数为压缩级别，0为不压缩，3为合适的压缩级别
dump(grid_search, 'grid_search.dmp', compress=3)
#从文件系统中加载数据到内存中
grid_search = load('grid_search.dmp')
```

6 回顾

包	类或方法	说明
sklearn.pipeline	Pipeline	流水线处理
sklearn.pipeline	FeatureUnion	并行处理
sklearn.grid_search	GridSearchCV	网格搜索调参
externals.joblib	dump	数据持久化
externals.joblib	load	从文件系统中加载数据至内存

注意：组合和持久化都会涉及pickle技术，在sklearn的技术文档中有说明，将lambda定义的函数作为FunctionTransformer的自定义转换函数将不能pickle化。

7 总结

2015年我设计了一个基于sklearn的自动化特征工程的工具，其以Mysql数据库作为原始数据源，提供了“灵活的”特征提取、特征处理的配置方法，同时重新封装了数据、特征和模型，以方便调度系统识别。说灵活，其实也只是通过配置文件的方式定义每个特征的提取和处理的sql语句。但是纯粹使用sql语句来进行特征处理是很勉强的，除去特征提取以外，我又造了一回轮子，原来sklearn提供了这么优秀的特征处理、工作组合等功能。所以，我在这个博客中先不提任何算法和模型，先从数据挖掘工作的第一步开始，使用基于Python的各个工具把大部分步骤都走了一遍（抱歉，我暂时忽略了特征提取），希望这样的梳理能够少让初学者走弯路吧。

8 参考资料

1. 使用sklearn做单机特征工程
2. FunctionTransformer
3. Github:jasonfreak/ali2015

分类: 数据挖掘

标签: Python, sklearn, 数据挖掘

jasonfreak
关注 - 0
粉丝 - 191
+加关注

17 0

« 上一篇：使用sklearn做单机特征工程
» 下一篇：关于线性模型你可能还不知道的二三事（一、样本）

posted on 2016-05-04 11:46 jasonfreak 阅读(42923) 评论(21) 编辑 收藏

Feedback

#1楼 2016-05-30 15:07 Michael_翔

期待博主的更新~

支持(0) 反对(0)

#2楼 2016-06-24 16:26 Lydon

好文。博主对sklearn理解深刻啊。
希望继续写。

支持(0) 反对(0)

#3楼 2016-07-27 23:36 fobdddf

楼主好文！
麻烦请教一个问题：
X_new = SelectKBest(chi2, k=2).fit_transform(X, y)
这种不是有监督的吗？

支持(0) 反对(0)

#4楼[楼主] 2016-07-28 10:43 jasonfreak

@ fobdddf
感谢您这么仔细的阅读！
笔误了，SelectKBest既可无监督，也可有监督，就看自定义的评分函数是不是需要利用目标值向量y的信息了
您说的评分函数微chi2的情况下，确实是有监督的。
已修正，再次感谢！

支持(0) 反对(0)

#5楼 2016-08-07 20:06 wangcq

博主，我在数据集上处理数据分类正确率只有70%多一点，如何才能提高分类正确率，达到百分之80以上

支持(0) 反对(0)

#6楼[楼主] 2016-08-08 08:03 jasonfreak

@ wangcq
您好，感谢关注！
首先您说的分类正确率是指哪一个指标？Accuracy、Precision、Recall、F1或其他？是交叉验证得来的，还是在测试集上预测得来的？
要让“分类正确率”有10个点以上的提升，我能想到以下方法：
1 特征方案：跟具体业务场景相关，非常重要，非常重要，非常重要！
2 预处理：首先，确定数据是否无量纲化了，如果是使用的线性核的支持向量机，还得确保样本归一化。
3 如果是只考虑正类的分类好坏（Precision、Recall、F1等），还需要考虑类别是否平衡，如果不平衡，一个最直接的方式就是对正类进行过采样，或者对负类进行子采样。
4 泛化：不要过拟合了，可使用交叉验证等手段。
5 调参：如果是使用的RF，其参数n_estimators的默认值为10，一般来说都太小，通过增大该参数，可以获得较不错的分类效果。
总的来说，只要不犯“低级错误”，最有效能提高“分类正确率”的还是特征方案这一途径。

支持(0) 反对(0)

#7楼 2016-08-12 22:19 丁磊-ml

好文，博主对特征工程理解好深啊！！
是因为哪本书有讲解吗？？？
也希望像博主那样，能把机器学习，特征处理弄的那么好．求博主推荐些资源，（比如书籍，视频，网站什么的），自己想系统研究这方面的知识．

支持(0) 反对(0)

#8楼[楼主] 2016-08-13 07:29 jasonfreak

@ 丁磊-ml

感谢关注，过奖了
要说系统地学习这方面知识，通常认为：首先得有一定的数学基础：概率论与统计分析，线性代数，凸优化理论。然后坚持上完（并且听懂）coursera上Andrew Ng的ML课程。接着在Kaggle、天池等找比赛练手，练手时边用边学习Scikit-learn、Weka等工具。在不断地实践与反刍中，一个新生的数据矿工就诞生了。

但是，因为各种原因（懒、英语不够好、个人偏好等）往往我们并不会走这条最理想的路线，哈哈哈。我个人的建议是先得打好概率论与线性代数的基础，然后找一本经典且对口味的教科书（或课程）以较快的速度学一遍，接着尝试一下自己实现某些模型与算法。再接下来，在比赛和项目中，使用机器学习工具（定一下来一个，集中精力去学习这一个），并且分析之前自己对模型与算法的实现与这些工具的实现之间的差异。每一次比赛和项目都要给自己定一个优化的目标（需求驱动学习），使自己对机器学习及数据挖掘有更深层次的理解。当实践遇到瓶颈时，回过头来再进一步夯实理论（凸优化理论，机器学习模型与算法），会发现自己对工具的使用有诸多错误的地方，甚至能够发现工具有更多可改进的地方，这时才真正打开了数据科学的大门。

支持(0) 反对(0)

#9楼 2016-08-13 09:08 丁磊-ml

@ jasonfreak
嗯嗯．谢谢大神！！！

支持(0) 反对(0)

#10楼 2016-12-27 22:31 魔灵幽亭

```
ValueError: operands could not be broadcast together with shapes (1,100) (100,3)
```

博主，你好！非常感谢你的文章！但是，我运行到网格搜索产生numpy广播错误。我感觉是，pipeline的step2_2每次只能操作一列（这样没有错误）。但是这样，进行网格搜索产生了一个inf错误，即

```
Input contains NaN, infinity or a value too large for dtype('float64').
```

对于这两个错误，不知是什么问题，我的sklearn版本是0.18.1,不知是否和博主一样？希望博主可以给点解决的建议。谢谢。

支持(1) 反对(0)

#11楼 2016-12-29 17:06 aoliong

@ 魔灵幽亭
我之前使用sklearn的0.17版时，没有出现这两个错误，但更新到0.18后，就出现同样的问题了

支持(0) 反对(0)

#12楼 2016-12-29 23:55 aoliong

FeatureUnion的实现在sklearn的0.18版中做了更新，接口参数与0.17版的不一样了。可以参考源码来修改就可以了运行了。

修改的地方主要是要增加weight参数，最简单的方法是将所有属性的weight设为1.

```
+ View Code
```

支持(0) 反对(0)

#13楼 2017-01-13 10:29 会飞的蜗牛

@ 魔灵幽亭
在你的数据集DataFrame上加一句df = df.fillna(0)

支持(0) 反对(0)

#14楼 2017-01-13 16:41 魔灵幽亭

@ aoliong
谢谢。

按照你所说的，我感觉博主的代码在0.18只需要将transformer_weights前置一下就可以了。如下：

```
delayed(_fit_transform_one)(trans, name, self.transformer_weights, X[:,idx], y,
                                        **fit_params)
```

之后确实都可以运行。
但是网格搜索报了警告：

```
DeprecationWarning: Estimator FeatureUnionExt modifies parameters in __init__. This behavior is depre
  % type(estimator).__name__, DeprecationWarning)
```

不知你有没有遇到。我看到这个以为scikit-learn有了更合适的方式来实现类似的处理方法。结果没有找到，只有一个issue在讨论如何解决。并且，没有理解报这个警告的原因。

Extend `FeatureUnion` to better handle heterogeneous data #2034

支持(0) 反对(0)

#15楼 2017-03-12 11:00 dhdsjy

不知博主现在是否还关注这个系列的博客？有个问题想请教您，使用流水线处理时，每个step都是对特征矩阵整个的进行处理吗？比如二值化，我可能只想对某一列进行二值化 流水线处理可以指定吗 好像官网上也没有这方面的具体介绍 博主能解决一下我的疑问吗

支持(0) 反对(0)

#16楼 2017-07-24 21:13 lchzh

你好，博主，我想查看使用sklaern训练出的模型文件的内容，有什么方法可以查看吗？在网查了很长时间，但是没有这方面的内容，只有模型的持久化。十分感谢能回答我这个问题。

支持(0) 反对(0)

#17楼 2017-09-16 14:46 xinqiyang

博主写的非常好， 现在在哪里上班啊。

真心非常赞，内容讲的都是干货。非常非常赞的说。

支持(0) 反对(0)

#18楼 2017-09-21 22:13 哈士奇说喵

牛掰！

支持(0) 反对(0)

#19楼 2017-09-28 16:04 kktree

非常不错！！

支持(0) 反对(0)

#20楼 2017-11-21 10:18 Rookie.

谢谢分享

支持(0) 反对(0)

#21楼 2017-12-14 17:15 liuer2009

@ 会飞的蜗牛

引用

@魔灵幽亭
在你的数据集DataFrame上加一句df = df.fillna(0)

支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论，请 登录 或 注册，访问网站首页。