常用特征工程方法

图灵：楼兰

> 上节课内容回顾

这一章我们就会进入非常有趣的部分，真正开始进行机器学习。课程开始要给大家说明下，机器学习这个领域，更多的是使用python语言或者R语言，而spark的java代码，反而会显得相对比较重。所以，在介绍机器学习时，会同时引入python和spark的代码给大家做演示，目的是为了让大家通过对比，也增加一点对于python尤其是sklearn框架的了解，这样，基于我们的课程，大家就可以去kaggle上转一转了。

四、机器学习-特征工程基础

1、机器学习处理什么问题

机器学习处理的问题通常都是一些规律不是特别明显的问题，这类问题通常很难通过简单的抽取规律来解决。举个例子，在公务员考试中，通常都会有这样的逻辑题：给你一串数字， 1,3,5,7,? 然后让你去推断下一个数字。或许这个题太简单，你知道要填个9，这就是一串奇数嘛。这只是一个简单的计算过程，换一种说法，这也就是一个预测的过程。通过已有的数据寻找规律，预测出下一个数字是什么样的。

> 有个小问题 如果有这样一串数字 1,3,4,7,9,11,？ 你觉得这个?是什么数字?

在机器学习中，最为典型的分类算法和回归算法，他们的处理流程也是类似于这样一个过程。通过对历史数据的推断，寻找数字之间的规律，从而预测出后面的数字是什么样的。只不过，他要处理的问题，比这个简单的数字推断更复杂。机器学习的历史数据不再是一个一个的数字，而是由多个数字组成的向量。并且，数据之间的规律更难找到，同时也没有这么稳固。很可能数字之间并没有完全准确的规则，这时就需要选择出一个相对靠谱的数字来。其实这个思想跟之前的数字推断是一样的。

> 机器学习处理的是数字向量的问题，如果特征值是图片要怎么处理？

2、机器学习的标准处理流程

一个典型的机器学习项目通常分为以下几个步骤：

1- 数据收集

首先需要收集到尽量多，尽量全的业务数据，这样整个机器学习的结果才更准确，更有说服力。这就好比我们经常说的一句话："所有人都是这么说的"。还是以我们那个逻辑题为例，如果只给你 1,3,? 你或许也能推测出下一个数字是5，但是这个规律就非常勉强了。而反过来，如果这一串数字给得越多，这个规律就会得到不断的验证，最终效果就会越明显，也越能处理实际的问题。例如，1,3,4,7,9,? 如果是这样的题目，本身的规律就变得非常模糊了。但是如果这串数字还按照奇数的规律不断重复，100个数字，10000个数字，这个时候，整个奇数串的规律是不是还是可以接受了？偶尔一两个数字的偏差也是在可接受范围内。其实我们处理实际问题也是这样一个过程，比如预测天气，我们通常会总结出很多的生活规律，比如天气特别闷热，晚上出现毛月亮，那我们就会推测明天会下雨。尽管这个推测很多时候是不靠谱的，但是还是会形成这样的整体规律。为什么呢？这就是因为有别人非常多的经验，这些经验就是来自于对历史数据的学习。

这也符合机器学习的工程要求。机器学习会通过学习历史数据，总结出一些最有可能的规律。当这些规律达到一个比较高的可信度时，就可以用来对未来数据进行预测了。所以数据的体量以及质量，往往就决定了机器学习所能达到的高度。这也是为什么很多好的机器学习产品最先都是出在像谷歌、百度这样的大公司的，就是因为他们的数据往往是最大最全的。

2- 数据清洗

然后有了数据之后还需要进行清洗。原始的数据就像是矿石，往往含金量非常低。这个时候就要通过数据清洗，将明显无用的信息去掉，并且把数据整理成能够被机器学习接收的数据格式。这个过程通常没有固定的工作方式，需要根据不同的算法不同的要求，指定不同的处理方式。数据清洗是前期工作量非常大的一个环节，同时也是非常考验程序员工程能力的环节。

2- 训练模型

这个过程是最关键的，但是其实他也是比较简单的。有了数据之后，你只需要选择一个合适的机器学习算法，把数据交给他学习，自然就会形成一个数据模型。这个过程往往不需要人工进行干预。甚至很多时候，机器学习到底学习到了哪些规律，人也是很难弄明白的。在这个过程中，需要注意的是，针对同一个问题，往往可以选择很多的算法，甚至针对同一个算法，也会需要制定不同的超参数。这些组合都会计算出不同的数据模型。所有这些模型都是可选的结果。

3- 模型优化

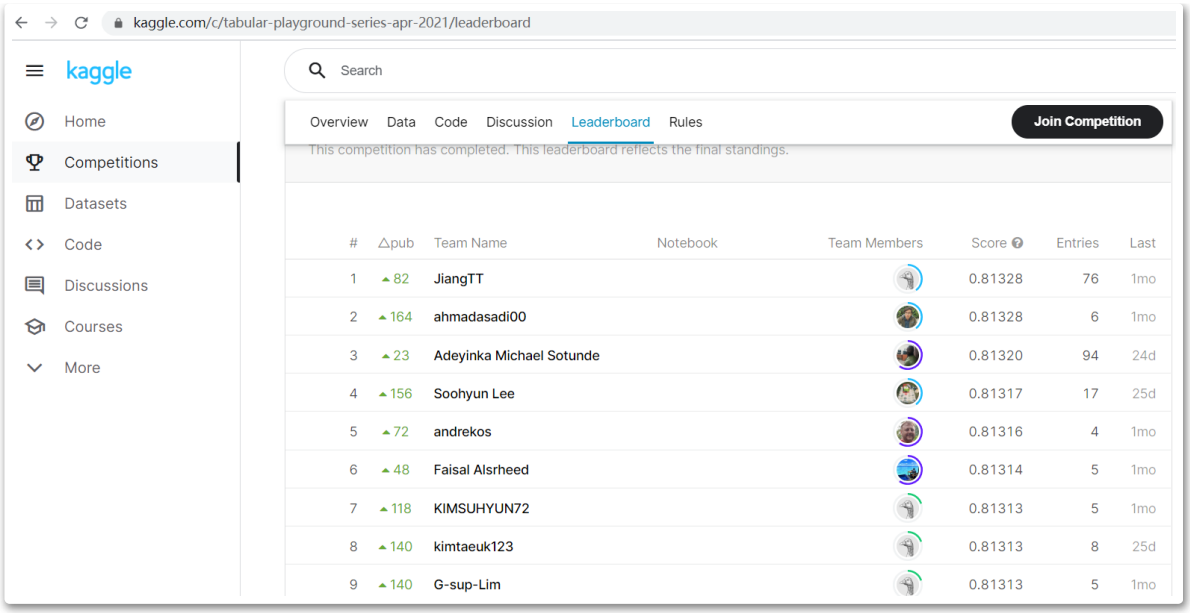
训练出了模型，并不能代表机器学习成功。有了众多的数据模型之后，就需要在这些模型中找出针对当前问题的最佳方案。这个优化过程即需要基于对算法的深入了解，同时也需要基于大量的尝试。也是非常考验算法工程师技术的地方。

最常用的检测方案是将整个数据集随机拆分成训练集和测试集。用机器学习算法在训练集上学习并形成数据模型，然后拿这个数据模型对测试集的数据进行预测，接下来拿预测出来的目标值与测试集上实际的目标值进行比对。目标值真实结果匹配度最高的模型就认为是最好的模型。我们经常说的人脸识别准确率达到多少多少，其实就是在测试集上的比对结果。

另外，从历史数据中学习形成的数据模型，最终还是要回归于对未来业务的指导，而未来业务又会形成新的数据集。这个时候模型优化一个很重要的过程就是需要让模型继续学习更多新的业务数据，及时优化。这样才能让模型的准确地更高。

3、为什么需要特征工程(Feature Engineering)?

我们首先看一个kaggle上的竞赛排名结果 https://www.kaggle.com/c/tabular-playground-series-apr-2021/leaderboard



> 这个榜单上有很多中国的学生

可以看到，针对同一个问题，同样的数据集，竞赛中却跑出来了非常多不同的成绩。那大家想一下，数据集是一样的，算法也就那么几种，这些顶级竞赛者肯定也都懂。那他们应该要跑出同样的结果才对。那为什么还能出现这么多结果，排出个名次来呢？这个区别就来自于我们这一章节要讨论的特征工程。

业界流传一句非常经典的话：**数据和特征决定了机器学习的上限，而模型和算法只是不断逼近这个上限而已。**那要怎么才能逼近呢？这就要靠特征工程。

特征工程是使用专业背景知识和技巧，处理数据，使得特征能够在机器学习算法上发挥更好的作用的过程。也就是说，**特征工程会直接影响机器学习的效率。**而这些各种机器学习竞赛，之所以分出不同的高下，很大一部分原因就在于他们的特征工程处理方式不同。

这里可以看到，其实特征工程也就是在数据收集清洗完成之后，在机器学习之前，对数据进行预处理的过程。这个过程，跟机器学习是没有太多关系的。也就是说，做特征工程，其实不需要机器学习的知识，你大可以随心所欲，想怎么做就怎么做。但是，如果你想要让后面机器学习的效果比较好，那有些普遍的工具和处理方法还是要懂的。

> 具体可以参见准备的python sklearn示例，以及spark的官方示例。

4、常用的特征工程方法

特征工程的方法很多，大致可以按照作用分为以下几种方式：

1、特征抽取

机器学习只能学习数字类型的特征值，但是有些数据集他的原始数据不是数字类型，比如图像、文本、字符等。这时，就需要使用特征抽取将数据转化成适合机器学习的数字特征。

1> 文本特征抽取：

场景：现在如果我们要对文本的分类情况进行机器学习，这是一个典型的分类问题，但是这个特征值似乎跟我们之前提到的特征值不太一样。文本没有那些属性和数字啊。那应该怎么抽取特征值呢？

1、one-hot编码

对于字典类型的数据特征，例如 性别、城市、颜色这一类字典类型数据，通常在数据库中，会以一个数字编码标识，如 0:男,1:女 这样。但是，如果在机器学习中使用这样的数字编码，就会给学习过程造成误解。因为不同的字典值特性应该是完全“平等”，而如果是 0，1，2这样的数字，则可能给机器学习造成误解，觉得这些字典值是有大小关系的。所以，机器学习中常用的方式是把字典值处理成one-hot编码。

性别	性别==男	性别==女	one-hot编码
男	1	0	(1,0)
女	0	1	(0,1)

> sklearn方法：sklearn.feature_extraction.DictVectorizer
>
> spark示例：OneHotDemo
>
> 实现效果是不一样的。

2、CountVectorize

对于文本类型的数据，如一篇文章。在做机器学习时，最基础的处理方式是以文章中的单词出现次数作为特征。处理成 [(word1,count1),(word2,count2)...]这样的格式。这也是mapreduce、spark最经典的入门计算方式。

> sklearn方法：sklearn.feature_extraction.CountVectorizer
>
> spark示例：CountVecDemo

缺点：在文章分类等机器学习场景中，体现不出文章的重要特征。例如一般出现次数最多的一些词，如，这里、那里、我们、他们等，并不能体现文章的内容特征。这类词称为停用词。

补充：
在对文本进行特征处理时，一般分词是个绕不开的坎。英文分词比较简单，按空格就行。而中文则麻烦得多。同时中文分词也是一个深度学习的重要研究领域，有一些可用的开源中文分词工具，例如java的hanlp，python的jieba等。使用开源的中文分词工具，很多的词语都要做特殊处理。例如如果你尝试去对鹿鼎记进行拆分，里面的很多名字、武功等就要做特殊处理，例如西奥图三世、白衣尼等。而对于一些更为专业的文章，这些开源的中文分词工具就都表现不是很好了。像百度的中文分词工具， https://cloud.baidu.com/product/nlp/lexical 大家可以上去体验一下。

3、TfidfVectorizer

我们的目的是为了对文本进行分类，但是简单的以单词出现的次数来分类，从经验上判断，会有些问题。长的文章中各个词出现的次数都会比较多，而短的文章各个词普遍都会比较少。这样长的文章对分类结果的影响就会被放大。这时改进的办法就是TF-IDF

TF-IDF可以用来评估一个字词对于一个文件集合或者一个语料库中的其中一份文件的重要程度。例如，在对一大堆文章进行分类时，出现 计算机、软件、云、java 这些词的次数比较多的文章更多可能归为科技类(在其他类中出现就比较少，这样的词才有重要性)，而出现 银行、信贷、信用卡 这类词出现次数较多的文章更多可能归为金融类。而所有文章中出现次数都比较多的 我们、你们、这里、那里等这一类的词则对分类来说，意义不大。

> · TF-IDF由TF和IDF两部分组成，
>
> - TF：词频 term frequency。某一个给定的词语在文章中出现的评率
> - IDF：逆向文档评率 inverse document frequency.是一个词语普遍重要性的度量。 为总文件数目 除以 包含该词语的文件的数量，再取10为底对数。
>
> 最终 TF-IDF=TF*IDF
>
> 示例：
> 关键词： “经济” ；语料库： 1000篇文章；10篇文章出现“经济” 。

> TF(经济) = 10/1000 = 0.01 ； IDF(经济)=lg(1000/10)=2
>
> 最终 TF-IDF(经济) = TF(经济)*IDF(经济) = 0.02

> sklearn方法：
> sklearn.feature_extraction.CountVectorizer.TfidfVectorizer
>
> spark示例：JavaTfIdfExample

2> 图像类型怎么进行特征提取?

图像类型的特征，最简单的特征提取方式，是按照图像每个像素点的RGB三信道值，抽取出三个信道矩阵，这样就有了基础的数学特征。但是图像特征的处理难度太大，机器学习基本处理不了。大都需要基于深度学习，神经网络来做。有兴趣的可以自己去了解下。

2、特征预处理

现在我们考虑这样一组特征值： 用户年龄和用户收入。我们能发现，年龄的数字相比收入的数字会小很多。根据之前特征要平等的原则，直觉上就会觉得，这一组数据集中，用户收入的特征会被放大，而用户年龄的特征就容易被忽略。

用户	用户年龄	用户收入
u_1	25	223780
u_2	40	323000
u_3	37	298000
u_4	47	525230

所以我们在用这样的数据集之前，要把各个特征值的范围尽量统一。这种方式称之为**无量纲化**，也就是要在保留数据之间的特征关系的同时，消除单位不同造成的特征之间的不平等。你可能会想，这组数据，我把用户收入除100，是不是也就把收入固定到了0~100的范围，跟年龄差不多？这确实也是一种处理办法，但是，这样的处理方法，一方面，量纲的影响还是没有完全统一，范围并没有填满。另一方面，用户收入这个特征的很多信息其实就丢失了，拿去机器学习就会丢失很多特征，效果就不会太好。那业界比较常用的方式有两种，归一化 和 标准化。

1、归一化

归一化通过对原始数据进行变换把数据映射到[0,1]这样一个标准区间。而这个标准区间是可以根据实际情况调整的。

他的标准计算公式如下：

$$X' = \frac{x - min}{max - min} \qquad X'' = X' * (mx - mi) + mi$$

> 其中max,min分别表示这一列特征值中的最大值和最小值。 而mx,mi表示指定的映射区间。默认mx为1，mi为0。
>
> 归一化的缺点：对异常值敏感。当数据集中出现一个不太合理的极大值或者极小值时，整个归一化的结果就非常不好。鲁棒性(稳定性)较差

> sklearn方法： sklearn.preprocessing.MinMaxScaler
>
> spark示例：JavaMinMaxScalerExample

2、标准化

前面提到，归一化对异常值是非常敏感的。而标准化就能很好的处理这种异常数据的问题。

标准化是通过对原始数据进行变换，把数据变换到均值为0，标准差为1的范围内。

他的计算公式是：

```
x'= (x-mean)/std
```

> mean： 特征值的均值， std：标准差。 均方差。
>
> 这种方式，对于极大或极小的少量异常点，均值和标准差都会比较稳定，所以异常值的影响就变小了。

> sklearn方法： sklearn.preprocessing.StandardScaler
>
> spark示例：JavaStandardScalerExample

3、降维

经过前面一通处理，大家是不是觉得数据变得越来越复杂了？特征值即变多了，也变得难看了。那怎么把这些数据简化一下呢？这就是后面的降维操作了。

这里的降维是什么？跟三体里的降维打击不是同一个东西。我们这里的降维是指在某些限定条件下，降低特征的个数，得到一组"不相关"的主变量的过程。那什么叫"不相关"呢？我们先简单的理解下什么叫特征与特征相关。例如，我们需要去学习某一个地区的降雨量，就会去统计一些常用的天气特征。而这其中，相对湿度与降雨量就是一个相关的特征，相对湿度大，肯定降雨量就会偏大。在进行机器学习训练算法时，如果特征本身存在问题或者特征之间相关性较强，那对于算法学习预测的影响就会比较大。而我们将为的过程，不光是要降低特征值的个数，同时也要尽量保留不相关的特征。

降维主要有两种方式：

1>特征选择

数据中包含了冗余或者相关变量时，通过数学方法从原有特征中找出主要的特征。选择的方式也有两个

- 方差选择法： 过滤低方差特征(过于集中的数据)

> 在sklearn中封装了一个函数:
> sklearn.feature_selection.VarianceThreshold 可以根据预设的阈值过滤数据集中的特征值。
>
> 在Spark中没有找到对应的示例，这里就不再去深究了。

- 相关系数法：特征与特征之间的相关程度。例如 天气湿度 与 降雨量 一般就认为是相关性很强的特征。

Spark可以查看示例：JavaCorrelationExample 。里面计算了皮尔逊距离和斯皮尔曼距离的矩阵。关于相关性，如果相关性为正数，表示两个特征正相关，即一个特征值越大，另一个特征值也越大。而如果是负数，就表示两个特征负相关。另外计算出来的关系系数的绝对值越大，表示特征的关联性更强。对于相关的特征，可以再考虑将他们组合成一个新的特征。

这个里面涉及到大量的数学计算，这节课就不给大家演示了。有兴趣大家可以课后深入了解。

2>主成分分析

定义：一种将高维数据转换为低维数据的方法。保留对目标值结果影响较大的特征值，去掉相对影响较小的特征值。

作用：数据维度压缩，尽可能降低原数据的维数(特征个数)，损失少量的信息。

理解：将高纬度数据投影到低纬度空间的过程。例如，一个(x,y)两维坐标点可以投影到一条y=f(x)的一维直线上，或者像皮影戏一样，将一个三维物体投影到一个二维平面上。而损失的信息量，可以简单的理解为能不能从低维还原成高维。比如照一张照片，从照片能不能推断出原来是个什么东西。 这个过程跟三体里的降维打击就有点相同了，只是是一种数学上的表达方方式。

高深的解释太复杂，最简单的理解就是特征个数太多，不好分析时，就可以用PCA来减少特征个数。

> sklearn方法： sklearn. decoposition.PCA
>
> spark示例：JavaPCAExample

总结

这一节课的目标是带大家了解机器学习领域一些耳熟能详的基础工程方法。结合下一节课的基础算法后，大家就可以上kaggle去逛一逛，看一些简单的例子了。这样在接下来设计推荐系统时，就能更加得心应手。

> 特征工程其实并不复杂，就是对数据进行处理。如果大家有兴趣想要更深入的了解下机器学习，给大家推荐一个kaggle上的入门教程： https://www.kaggle.com/ialimustufa/titanic-beginner-s-guide-with-sklearn 。你只需要简单的了解下Python的基础语法，这整个过程还是可以跟下来的。感受下简单的特征工程是怎么做的。如果你能够把整个过程跟下来，可以再尝试把我们这节课的特征工程算法给加上去试试。
>
> 特征工程却是机器学习最为重要的工作，也是机器学习算法工程师最经常做的事情(还有一个就是调算法的参数)。而对于Kaggle上的机器学习竞赛，拉开差距的地方也就在特征工程。好的特征工程能够让机器学习算法发挥更大的效果，更加逼近特征与数据带来学习效果的上限。这一

> 节给大家介绍的几种特征工程算法是经过很多机器学习过程验证过之后，业界普遍认为效果比较好的特征工程处理方式。
>
> 而通过这一节课，我们绕开了那些复杂的基础知识，直接带大家体验了一下特征工程是怎么做的。并且，其实在机器学习领域，大家应该能够发现，java语言还是太重了，所以一般还是python或者R语言这种直接一上来就做事的语言跟更适合。但是这节，也带大家了解了Spark中对机器学习算法的相关封装。再结合我们后面的大数据体系课程，相信机器学习离我们大家并不是太远。

> 文档：C2_常用特征工程方法.md
> 链接：http://note.youdao.com/noteshare?id=cbf150f8f55bcfbf7bc6ca9a7c1880d5&sub=A958FB606EA74B358FB873D20CAB36CF