

你知道我使用Matlab的心情吗？

最近为了上一门课我使用了Matlab，我负责的是数据科学哪些部分，主要就是机器学习那些，传统的，神经网络都有。然后我发现，Matlab真是太有个性了，把我折腾的死去活来的，所以我在这里给大家的结论就是，如果你做机器学习不要使用Matlab。



JUST KILL ME...

Kill me now...

首先，搞个刚刚开始用的时候我还觉得很爽，他的数据导入，和table特别好用，table的field还带自动补全，简直是太贴心了。画图也比较方便，基本的图基本上一行就能画出来。虽然Matlab里面的index是从1开始的，而且选择的时候还包含最后一个index，非常反人来，但是习惯了也是可以接受没有任何问题的。不过也有个麻烦就是他的indexing居然是()而不是[]。()和函数调用是重复的，好不爽，然是这些都是可以接受的。

然后剩下的全是要你命的东西，不过我可以总结成一句话，就是**inconsistent**！

首先各种模型，来自于不同的工具箱，他们的命名规则都不一样，玩法套路也都不一样。

例如有的模型是 `model=fit(x,y)` ,然后返回的模型inference的时候是 `y=model(x)` 。你要问, 这fit的是个啥模型呀? 我就呵呵了, 注意这个fit不是fit某个模型, fit就是个模型, 一个线性模型, 你给sample和option, 它返回一个model。

然后有个叫fitlm的, 这个倒是好象和上面一样, 他是fit另外一个叫lm (linear model) 的模型, 这两个是在不同的工具箱里面。但是如果你天真的认为他们是一样的你就完蛋了, 因为根本就不一样!!! fit返回的是一个model, 你通过 `model(x)` 来inference, 但是fitlm返回这个模型却是用 `model.predict(x)` 来inference的, 为什么要搞不一样?

然后更匪夷所思的是mnrfit, 你猜这个是个啥? 这是multi-nominal logistic regression按照上面这个套路, 为啥不叫fitmnr, fitmnl? 关键是你猜它返回的是什么? 不是模型, 完全不是它返回的不是object, 是一个系数矩阵!! 我阵亡了。。。一个系数矩阵怎么搞? 怎么做inference? 好, 我硬核, 我可以推导出公式, 然后我发现他很反人类的再logistics regression公式里面加了个负号, ok这些我从文档都推得出来。

但是让人吐血的是, 是整个文档, 整个文档里面, 没有教你怎么做inference, 直到, 最下面有个also see里面有好几个函数, 其中写了个函数叫mnrval, 这是干啥的? 我为什么要点开他。当你点开他以后你会惊奇的发现, 这个函数是用来做上面这个mnrfit模型inference的。我猜他是multi-nominal logistic regression value的做缩写, 不过不应该叫evaluate吗, 这怎么起的名字(这真不是重点)? 重点是要用这个系数做inference你要这么操作: `pihat = mnrval(B,X)` , B是之前的系数矩阵, X是sample。又是一种新玩法!

然后我们再看另一个模型, `Mdl = arima(2,1,2)`;这是啥? 按照前面的玩法, 这应该返回的就是一个fit过的模型吧, 那sample怎么给进去的, 没有呀? 呵呵, 这个模型根本就没有fit过, 这里返回的是一个初始化的模型, 所有参数都是默认值。你通过 `model=estimate(Mdl,X)` 来fit这个模型。what! estimate是什么鬼。对就是fit, 他起了个更有个性的名字。然后你有了fit后的模型怎么预测能, 我告诉你: `forecast(model,15,X)` 。你看, 是不是又获得了新知识?

再说收statistic and machine learning toolbox, 也不一样。首先和前面线性模型有点像, 同通过 `fitsvm, fittree` 这些方法来得到一个模型, 但是预测又不一样了, 注意这里面有的模型是 `pridict(model,x)` 有的是 `model.predict(x)` , 有的是 `model.predictFcn(x)` 。你看我排比句都用上了。

然后再说说matlab的神经网络, 不是deeplearning哈, 就是shallow神经网络工具箱那些, neural net fitting那些。简直不能忍, 首先, 他的输入时列向量为sample, 但是之前其他的都是行向量。然后他使用train这个函数来训练, 不是之前的fit, estimate那些, 然后interface的时候时 `y=model(x)` 这种方法!!!

然后同一个工具箱里面, 那个NARXNET, 那个时间序列的网络模型, 他又不一样。首先, 他的输入时cell不是matrix, 文档里没有, 他妈我研究了好久才发现。然后这个网络只有两个输入, 就是X, Y。那X, Y有多个特征怎么办, 对, 用cell。他把所有的特征放在一个cell里面, woc, 这想象力真丰富。别

问我怎么发现的，我搞了2小时。然后我想把矩阵变成等尺寸的cell，怎么搞？mat2cell，想当然就是做这个的，但是他是切割矩阵的，然后我研究怎么给参数让他正好一个元素一个坑，倒是可以就是很慢，直到我发现了num2cell，这个瞬间可以完场上面的操作，WTF？这个2cell应该是2 an element of a cell array,其实时make sense的，但是only make matlab sense, not everyday sense。然后我发现了，卡面说的所有的特征压缩到一个cell里面，我崩溃了，好吧，在转换一下，直到我发现了，再他声称的代码里面有个现所，有个叫tonndata的函数，直接可以把矩阵处理成这个网络需要的cell数组。擦。

你以为就此结束了，不不不。那个网络怎么inference？不是 $y = \text{model}(x)$ ，由于他是个自回归的时间序列网络，你要给他以前过去的值才行，不仅要给他过去的值，还要给他这个网络通过过去的事积累的一个状态。

```
[Xs,Xi,Ai,Ts] = preparets(net4,xs_test(1:24),{ },ys_test(1:24));  
yhat = net4(xs_test(25:end),Xi,Ai);
```

第一个函数唯一有用的就是输出Ai，就是网络初始状态，是由Xi确定的，Xi就是我自己给的，完全不需要他帮我准备。你看，第一个函数给了net，后面一个也有net其实完全可以把preparets放到inference的代码里面嘛，inference的时候自己计算一下初始状态就行了。有人要说这个functional programming的style，net是immutable的，所以要外界给他初始状态。听起来有理实际上放屁。你完全可以interface的时候计算了初始状态，然后扔掉吗。对了之前将得分ARXM那个模型，同样是时间序列就不需要自己整这些。

然后再说说他的deep learning，他的玩法不一样。首先你设计好一个网络以后是通过trainNetwork来训练的，好，又不一样。然后inference的时候更奇葩，他有两个

个 $YPred = \text{classify}(\text{net}, X)$ 和 $YPred = \text{predict}(\text{net}, X)$ ，不仅和前面的不一样，自己居然都不一样。不过好象也能理解，predict是直接网络的输出，classify是吧网络输出的onehot编码转成了categorical的数据。不过这都不重要，因为这个根本就没法用，他的网络输入特别固定，只有图像和序列，也就是说输入最少是3维的，这3维怎么flatten的？没人知道。我想给个一维的特征就用了 $1 * 1 * n$ 的或者是 $n * 1 * 1$ 的，应该可以把，结果loss纹丝不动就是不降，不知道为啥。同样的模型再python keras里面一次成功。所以千万不要用matlab的deap learning。

Matlab还有很多很多不一致的地方，有很多模型可以接受表格和matrix，有的只能matrix，有的只能cell，总时折腾死你。为啥不能和python sklearn或者TF那些学习一下，nparray首先都可以输入，然后就一套 model.fit model.predict，统一一点不可以吗？

一点最基本的写代码的素质都没有。

当然Matlab优点还是很多的，只要你把那他来做数据科学，他用起来还是很顺手的。