# 待解决的问题

## 选择epoch大小

见pyml495，通过我已经做的图判断epoch什么时候最好（两曲线重合），从而避免过拟合。

## 看自己的模型有几层

看看自己的模型到底是几层，貌似输入层和第一层隐藏层会合并设置（第一层的节点数不需要等于输入特征数）（pyml书的p535貌似有讲），那么实际keras设计层数时只需要两句add语句。参考网址：<https://blog.csdn.net/fang_chuan/article/details/88559930>

答：

根据我目前研究，输入层到底算不算层数有争议（因为其没多大作用），但是一般讨论隐藏层的层数就行了。一般来说，一层隐藏层可以解决很简单的分类问题，两层差不多了就够用了。

## 评估模型

使用k折线可以评估模型

## prediction功能

据pyml p82，可以model+其他井的x=其他井的y-predic。然后把y-predict存下来，对比其他井的y

## 逻辑回归模型

用pyml中的逻辑回归范例去尝试实现一下。比较他和神经网络两个模型之间的准确度。

# 已解决的问题

## 解决正负样本不均衡的问题

想想怎么解决正负样本不均衡的问题。就是我把预测全部设置成不溢流，得出的准确度反而可能更高。这时候就不能只是参考准确度了，还要参考 回收率等。具体内容见pyml

答：

我记得pyml中说了，最好的时候就是数据集中，正负样本各占一半。在学长的代码里，已经只使用了前后一天的negative data，我看positive data的占比基本已经到了1/3，差不多够了。他是选取了临近一天的数据后，复写原来的数据。

## 保证测试集和训练集中正负标签比例相同

仿照pythonml的332页前面的某个部分（p81），把两个label的数据先分开，再分别对两个label去分割出训练集和测试集。这样可以保证训练集和测试集中的两个label的比例相同（但是先弄清，train split函数会不会就y去等比例分割）

答：

其实不需要就y分割，因为split是随机的，所以比例是相同的。但是此时我们就得保证溢流的数据在总数据中的占比要比较大（跳到1.3）。

## 神经网络种子问题

1. 神经网络为什么需要种子
2. 我的神经网络的种子设置了吗？是什么？

答：

1. Artificial neural networks use randomness while being fit on a dataset, such as random

initial weights and random shuffling of data during each training epoch during stochastic gradient descent.

网址：https://machinelearningmastery.com/evaluate-skill-deep-learning-models/

Fix the Random Seed介绍了随机种子保证每次结果一样

How Many Repeats?（这里的repeat不是epoch）(即要设计多少次随机种子，来得到不同的模型)

I would recommend at least 30, perhaps 100

机器学习很多地方都会用到随机性。在train-test-split中的随机种子是为了随机分割数据。在tensorflow中的随机种子，如tf.Variable(tf.random\_normal([3,3], stddev=2, seed=1))是为了保障每次每次生成的随机数（集）都一样。

1. 设置了，train-test-split函数里面的random\_state就代表了这个函数专属的随机种子。

# 学习中的知识备注

## epoch和repeat的区别

<https://machinelearningmastery.com/evaluate-skill-deep-learning-models/>

问：How is running trainings times on same dataset different than running training through same data with multiple epochs ?

Can I say that by doing multiple epochs of data with deep learning serves the same purpose to reduce variation in results due to the stochastic nature of the Deep learning algorithm.

答：Jason Brownlee June 2, 2017 at 12:44 pm #

Each epoch updates the network weights.使得模型误差更接近局部、全局最小值

Each run through all epochs results in a different model given different random initial conditions（而每个随机种子对应一个model）.

Does that help?

## epoch和batch区别

<https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/>

1. When all training samples are used to create one batch, the learning algorithm is called batch gradient descent. 想起了吴恩达视频，gd是batch gradient descent的简称就是因为每轮回完所有的样本才会在jmin上走一步（更新参数）
2. When the batch is the size of one sample, the learning algorithm is called stochastic gradient descent.（吴恩达视频提过，这个在ml中性能更好，因为不用每次更新参数都对所有数据做一个复杂操作，而是那个操作总共只要做一次）
3. 综合上面两条我们得到：

Batch Gradient Descent. Batch Size = Size of Training Set

Stochastic Gradient Descent. Batch Size = 1

Mini-Batch Gradient Descent. 1 < Batch Size < Size of Training Set

1. 选择epoch的大小

(链接到1. 2）

The number of epochs is traditionally large, often hundreds or thousands, allowing the learning algorithm to run until the error from the model has been sufficiently minimized. You may see examples of the number of epochs in the literature and in tutorials set to 10, 100, 500, 1000, and larger.